DOCTORAL THESIS

Predicting Connectivity in Wireless Ad Hoc Networks

Larkin, Henry

Award date:
2005

Predicting Connectivity in Wireless Ad Hoc Networks

Henry Larkin
BIT (Hons)

A dissertation submitted in fulfilment of the requirements of the degree of Doctor of Philosophy for the School of Information Technology, Bond University

1st November 2005

Statement of Originality

The material in this thesis has not been previously submitted for a degree or diploma in any university. To the best of my knowledge this thesis contains no material previously published or written by another person except where due acknowledgement is made in the thesis itself.

Henry Larkin

Date:

Summary

The prevalence of wireless networks is on the increase. Society is becoming increasingly reliant on ubiquitous computing, where mobile devices play a key role. The use of wireless networking is a natural solution to providing connectivity for such devices. However, the availability of infrastructure in wireless networks is often limited. Such networks become dependent on wireless *ad hoc* networking, where nodes communicate and form paths of communication themselves. Wireless ad hoc networks present novel challenges in contrast to fixed infrastructure networks. The unpredictability of node movement and route availability become issues of significant importance where reliability is desired.

To improve reliability in wireless ad hoc networks, predicting future connectivity between mobile devices has been proposed. Predicting connectivity can be employed in a variety of routing protocols to improve route stability and reduce unexpected drop-offs of communication. Previous research in this field has been limited, with few proposals for generating future predictions for mobile nodes. Further work in this field is required to gain a better insight into the effectiveness of various solutions.

This thesis proposes such a solution to increase reliability in wireless ad hoc routing. This research presents two novel concepts to achieve this: the Communication Map (CM), and the Future Neighbours Table (FNT). The CM is a signal loss mapping solution. Signal loss maps delineate wireless signal propagation capabilities over physical space. With such a map, connectivity predictions are based on signal capabilities in the environment in which mobile nodes are deployed. This significantly improves accuracy of predictions in this and in previous research. Without such a map available, connectivity predictions have no knowledge of realistic spatial transmission ranges.

The FNT is a solution to provide routing algorithms with a predicted list of future periods of connectivity between all nodes in an established wireless ad hoc network. The availability of this information allows route selection in routing protocols to be greatly improved, benefiting connectivity. The FNT is generated from future node positional

information combined with the CM to provide predicted signal loss estimations at future intervals. Given acceptable signal loss values, the FNT is constructed as a list of periods of time in which the signal loss between pairs of nodes will rise above or fall below this acceptable value (predicted connectivity). Future node position information is ideally found in automated networks. Robotic nodes commonly operate where future node task movement is developed and planned into the future, ideal for use in predicted connectivity. Non-automated prediction is also possible, as there exist some situations where travel paths can be predictable, such as mobile users on a train or driving on a highway. Where future node movement is available, predictions of connectivity between nodes are possible.

Detailed analysis of the two proposed concepts are presented in this thesis. Comparisons with existing prediction algorithms illustrate that employing a signal loss map (the CM) vastly improves the accuracy of predictions. The fundamental concepts of the FNT are validated, though in the testing environment the FNT is not shown to be the ideal predicted connectivity architecture for wireless ad hoc networks in comparison to previous work.

Acknowledgments

This PhD thesis would not have been possible without the support of many people and organisations to whom I owe a great deal. I would like to thank particularly:

Dr. Zheng da Wu, for his belief in my abilities, his advice, his encouragement, his wisdom, and his endless kindness. You have been a great and wonderful supervisor.

Dr. Warren Toomey, for his years of supervision, suggestions, guidance, teachings and his willingness to listen when I needed to talk about everything technical.

The School of Information Technology, Bond University, for providing me with a scholarship, resources and the opportunity to complete a PhD, as well as introducing me to and allowing me to pursue my teaching career. Also thanks to the numerous staff over the years who have helped make life easier. Special thanks to *Stephanie Patching*, who continually made all the difficult paperwork that much easier.

The Australian Government, for providing me with an Australian Postgraduate Award during my PhD candidature.

To fellow PhD candidate *James Montgomery*, whose numerous donations of advice were outweighed only by the humour and reassurance he constantly provided. So much of this thesis would not have been possible without him.

Officemate *Helen Kennett*, who has been as much an inspiration as a companion in this journey of life. I doubt I will find an officemate as important as you have been to me.

Simon McGuinness and his family, for encouraging me, for providing me with food, comfort and internet, and making me feel like I had a second home.

Satomi Hagiwara, without whom I probably would have lost focus so long ago. I extend countless thanks for all the memories I've shared with you.

To the *numerous friends* who have helped make my life really mean something, for providing me with near infinite hours of enjoyment, direction and stability in life. Incredible thanks to *Watcharin Surapongwanitchakool*, for always believing in me. To Lina Piao for nudging me into life after PhD. To *Nick Tran*, *Laurent Corgnet*, *Paul Chen*, *Esther Choong*, *Maki Nonaka*, *Yukari Okano*, *Nick Hamilton*, *Takeshi Ashenden*, and the entire *Anime club*.

The many, many *students* who I have taught since I began teaching 7 years ago. It is because of you and the positive experiences that you have provided me with that have been the source of much inspiration over my entire life at Bond.

And finally *my family, Brian, Judy, Andrew, James and Kate*, for giving me the best home and family I could have hoped for, a place I never want to leave, a place I have been able to achieve everything from my education to my dreams & goals. Thank you.

Publications Arising from this Research

Larkin, H., Wu, Z., Toomey, W. (2004). *Mapping Wireless Signal Strength for Mobile Adhoc Networks*. Proceedings of the International Conference on Wireless Networks (ICWN'04), Las Vegas, USA, CSREA Press, pp. 151-157.

Larkin, H. (2004). *Wireless Signal Strength Topology Maps in Mobile Adhoc Networks*. International Conference on Embedded and Ubiquitous Computing (EUC'04), Aizu, Japan, Springer-Verlag, pp. 538-547.

Larkin, H., Wu, Z., Toomey, W. (2005). *Predicting Network Topology for Autonomous Wireless Nodes*. European Wireless 2005 (EW'05), Nicosia, Cyprus, VDE-Verlag, pp. 413-417.

Larkin, H., Wu, Z., Toomey, W. (2005). *Algorithms for Predicting Node Connectivity in Wireless Ad hoc Networks*. International Conference on Wireless Networks (ICWN'05), Las Vegas, USA, pp. 475-481.

Larkin, H., Wu, Z., Toomey, W. (2005). *Validity of Predicting Connectivity in Wireless Ad hoc Networks*. Proceedings of The International Conference on Mobile Ad-hoc and Sensor Networks (MSN'05), Wuhan, China, Springer-Verlag, pp. 643-653.

Larkin, H., Wu, Z., Toomey, W. (2005). *Performance of Signal Loss Maps for Wireless Ad hoc Networks*. Proceedings of IFIP International Conference on Embedded And Ubiquitous Computing (EUC'05), Nagasaki, Japan, Springer-Verlag, 609-618.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

Wireless networks are a growing area in both research and industry. The practicality and simplicity of operating without wires is expanding throughout the globe. In a multitude of environments and situations, laying a cabled network becomes either impossible or impractical, requiring alternatives such as wireless networking.

Wireless networks may be divided into two designs, fixed infrastructure and ad hoc wireless networks. Fixed infrastructure networks have a number of advantages, chiefly that they are reliable and managed. Fixed infrastructure networks, which include mobile telecommunication networks, are less likely to suffer changes in network topology [PB94]. They are also easier to manage and create algorithms for as they are usually designed for a specific purpose, and often centrally controlled. A single organisation often creates and maintains the network for this specific purpose. Thus the design of algorithms in relation to the hardware of devices becomes easier, as the organisation decides if and when hardware changes will occur. There is also no requirement for devices to compete for various roles in the network (such as cluster or router heads common in wireless routing protocols [CDT02] [KV98] [PC97] [JM96] [PB94]), as the role of devices is specified by the designers and administrators. For example, the role of a mobile phone will always be as a client device, whereas a mobile phone tower will always be a tower. However, because of these reasons fixed infrastructure networks are also limited and the network is unable to neither expand without hybrid technologies nor adjust to environmental changes.

Wireless ad hoc networks operate without a managed infrastructure, in contrast to fixed infrastructure networks. Ad hoc wireless networks have become a popular medium for networking, due largely to the simplicity and practicality of setting up such networks in real-world scenarios [Su00]. Ad hoc networks are created through the cooperation and sharing of any available nodes. This has the disadvantage that service guarantees cannot be provided, as well as the increased difficulty in managing nodes which act independently and outside the control of any central algorithm or process.

Because of these difficulties, research in this field has become a popular topic. The number of factors that must be considered and the unpredictability and unreliability of using radio as a transmission medium increase the conceptual difficulty of a single perfect solution. The different scenarios to which ad hoc networks may be applied leaves room for protocol specialisation. The technologies and network algorithms behind these networks are more difficult than with fixed infrastructure networks and are far more difficult than their wired counterparts.

In order for nodes to route traffic to achieve required network connectivity, some nodes must act as routers, forwarding packets through other routing nodes until a required destination node is found ([JM96]). In the majority of wireless ad hoc networks, it is not common to find any special "router-preferred" nodes, but rather all nodes are considered equal and each may become a routing node. However, there is no guarantee a node will stay connected or stay within communication range of other nodes, as both the wireless signal environment and the actions of the node or the node's controller (such as a human carrying a mobile phone as a node) are beyond the control of routing protocols. It is for these reasons that ad hoc wireless networking continues to be a field of research.

The overall aim of most wireless ad hoc routing protocols is to achieve consistent connectivity. A group of random nodes need to stay connected to the overall network at all times, such that every node is reachable by any other node. To achieve the best connectivity, this involves *handing over* connections (often called "handoffs") between a source and a destination from a route which is soon to disconnect to a route which has connectivity. Whereas in fixed infrastructure networks the infrastructure can hand off connections from one router to the next efficiently (such as antenna towers in mobile telecommunication systems), ad hoc networks cannot. Ad hoc networks do not have nodes appropriately distributed so that physical space is divided conveniently. Ad hoc networks also do not have fixed towers that will always remain stationary. The routing nodes in ad hoc networks may be constantly moving, making it difficult to hand off a connection through one router to another, where the replacement router must be selected from a group of perhaps inconveniently-placed nodes. There is also an issue of a routing node and a transmitting node moving out of range of each other, creating a connection loss for the transmitting node until it establishes a new connection through a routing node or nodes. While selecting preferred nodes as routers can be achieved by

extending existing routing algorithms, predicting when a change in network topology will occur is an issue unique to wireless ad hoc networks.

By predicting when nodes will lose a connection or gain a better connection between each other, routing algorithms can make decisions and reduce the chance of sudden changes in network topology. If handoffs in ad hoc wireless networks can operate as smoothly as those in fixed infrastructure networks, consistent connectivity can be achieved. However, wireless ad hoc routing protocols suffer from frequent changes in node mobility, resulting in reduced connectivity [RT99]. By being able to predict connectivity between adjacent nodes, existing routing protocols can be enhanced to increase connectivity.

Prediction of wireless connectivity consists of two parts. The first part is the prediction of node mobility. Many mobile nodes exist in scenarios where future movement is unknown, such as mobile phones in the possession of humans. However, by analysing the past movements of nodes in the same location, it has been possible to predict to an extent the future node mobility. Prediction accuracy can increase even further where a node's mobility is known. Cases exist where a mobile node's user is driving along a highway, taking a bus between cities, or riding a train, for example. A number of scenarios exist where user-carried nodes or even autonomous nodes move along known paths, resulting in increased accuracy of prediction. Robots that plan their own physical movements and communicate such information with surrounding nodes in the network are an example of this. If robots are given some task, such as mapping out a university, their movements can be organised and distributed. Tasks may change over time, but even a few minutes of predicted direction and speed is enough for routing algorithms, so long as the information is shared before it is needed. Node mobility can often be predicted with remarkable accuracy, given the right circumstances and environment.

The second part of predicting wireless connectivity relates to that of the wireless signal propagation environment. The range that a signal will travel in any one direction given a specific output power cannot be computed with accuracy without knowing some information about the wireless environment. Atmospheric and weather conditions, other transmitting devices, obstructions preventing a clear line-of-sight between nodes all affect how a wireless signal propagates through the atmosphere. Because of this, the

range that a wireless signal will travel can vary greatly, even over small areas. For example, if two nodes are across a street from each other, a travelling bus which blocks a clear line-of-sight between them will significantly reduce the signal's strength at the receiver end. A node going in and out of an office block, even if going inside only a few metres of travel, can cause a signal to drop off completely depending on the building's materials.

This thesis covers the topic of prediction in wireless ad hoc networks, more specifically where node mobility is known but where the wireless signal environment is unknown. This is used to provide a more accurate prediction of connectivity between nodes, which can then be applied to various ad hoc routing protocols to improve connectivity.

This thesis makes four primary contributions to the field of prediction in wireless ad hoc routing protocols:

1. A real-time, distributed signal loss map solution has been developed (the CM).
2. A novel approach to providing prediction information to routing algorithms has been developed (the FNT).
3. The performance of several approaches to adding prediction to on demand and distance vector-based routing protocols is analysed.
4. An extensive simulator, WANS, has been developed for the purposes of analysing performance of signal loss maps and prediction in wireless ad hoc networks.

In Chapter 2, existing research in this area will be summarised. In Chapter 3, the objectives of this research and the overall architecture of the solution will be presented. In Chapters 4 and 5, the two primary components of this research will be presented, including all algorithms required for the solution. Chapter 6 will investigate existing routing protocols, and how the prediction solutions in this research can be applied to them. Chapter 7 will overview the simulation environment used to test the concepts presented in this thesis. Chapter 8 will detail the results of all experiments, with Chapter 9 concluding this work.

Chapter 2: Routing in Wireless Ad hoc Networks

The area of prediction in ad hoc networks is relatively unexplored, and to date, no significant research exists which attempts to use *signal loss maps* to predict future node communication capabilities. That is to say, little research has tried to use knowledge of the wireless environment to inform routing decisions. However, a number of authors have attempted various aspects which, if combined, would make up such a solution. These are based on prediction, signal loss maps, and routing. In this chapter a literature review of these areas is given.

2.1 Predicting Wireless Connectivity

A large amount of research has been devoted to solving problems in the wireless ad hoc environment. The majority of these algorithms have focused on routing in the current environment, where network topologies are created with what is known currently about node connectivity. Somewhat less research has also ventured into using prediction as a means of better designing such wireless topologies.

The most significant contribution to prediction in ad hoc routing to date has been the work by William Su in his dissertation "Motion Prediction in Mobile/Wireless Networks" [Su00]. The work investigates the use of mobility prediction in improving connection quality in cellular networks. In particular, various existing ad hoc routing protocols are extended to include Su's mobility prediction proposal.

The significance of Su's proposal to this research is in the use of future knowledge. This is in contrast to the vast majority of alternative mobility prediction methods, which focus on using existing positional and movement data to predict future movement patterns. Su's work assumes that the current direction, speed and duration of travel in that direction are obtainable. This could be through examples such as GPS tracking in a car with wireless capabilities combined with a street map, or a user riding a train on a specific

track. Given a more accurate prediction of immediate future travel, neighbouring nodes can propagate such information so as to create a situation where all nodes know the immediate predicted travel plans of neighbouring nodes (it is assumed in the work that these predictions will be accurate).

Empowering nodes with this knowledge allows them to create Link Expiry Times (LET) for each node-neighbour pair, which is the estimated time when the link between the two nodes is predicted to terminate, given the current travel movement of both mobile nodes. Su applies this to various distance-vector, on-demand, and multicast routing protocols and demonstrates a greater packet delivery ratio in simulations than existing protocols LAR [KV98] and WRP [MG96]. The LET concept is a significant step forward in using node movement with routing protocols.

One issue not addressed in Su's thesis is the creation of the LET. The algorithm used for this calculation assumes a perfect *free-space loss* environment for all wireless signals between nodes. Free-space loss is the propagation of a signal in an environment free of obstructions, such that signals are neither amplified nor degraded beyond the natural signal loss due to propagation in all directions. In the real world it is almost impossible to find such perfect conditions, due to the atmosphere, surrounding structure (both environmental and man-made), weather conditions, and many more conditions which affect signal propagation. A signal loss map applied to the LET algorithms would have a significant impact on accuracy and performance of Su's design.

Su's work, as with all other known mobility prediction-based ad hoc routing algorithms, also overlooks autonomous nodes (such as robots) and fixed-path nodes (such as users travelling on trains). These nodes do not simply predict their current direction of travel, but have knowledge of their future movements for an extended period into the future over a course of multiple directions and speeds. With this information, far better LETs could be calculated.

Turgut, Das, and Chatterjee [TDC01] proposed a similar solution to [Su00], by exploring the expected lifetime of a route. The prediction of the lifetime of a route is possible where the destination node knows the location and velocity of all nodes along the route, the same as [Su00]. [TDC01] uses route predicted lifetimes to improve connectivity and

reduce bandwidth associated with route upkeep. In on-demand routing algorithms (such as DSR [JM96]) the route discovery process can be performed only when a route is predicted to break, ensuring connectivity. Predictions are based on four mobility models: deterministic, partially deterministic, Brownian motion and Brownian motion with drift. For each of the mobility models a prediction is obtained based on node speed and direction of travel. As with [Su00], the effects of the signal propagation environment are not discussed, nor is any signal loss map proposed. The authors note that a completely deterministic mobility pattern is highly improbable, and do not consider autonomous nodes.

The work by Chellappa-Doss, Jennings, and Shenoy [CJS03] has aimed to improve on Su's work by incorporating sectorized clusters. Where Su's work is constrained by a fixed speed and direction of each node, [CJS03] group nodes as being in a sector of a cluster. As nodes move to the outer limits of a cluster, the sector to which they belong also changes. These sectors are classified as No Cluster Change (No-CC), Low Cluster Change (Low-CC), and High Cluster Change (High-CC). These are defined such that nodes in the centre of a cluster are designated in the No-CC sector and nodes on the far limits of the sector belong to the High-CC region, where it is more likely those nodes will need to switch to another cluster. By keeping track of the sector a node exists in, improvements in predicting handoffs has been achieved. Their simulations showed that between 40% and 100% of handoffs were predicted However the work is limited to clusters only, and like Su's work could have achieved better accuracy if the sectors were not a fixed size, but based on the actual wireless signal environment.

An example of a more traditional approach to group mobility can be seen in work by Wang and Li [WL02]. In this research the authors predict changes to network topology based on current statistics in the network. This is in contrast to [Su00] and [CJS03], where future node mobility information is used. The focus in [WL02] is on predicting network partitioning before it occurs and centres around the idea that in real-world scenarios nodes often move in groups. These groups are given a mean group velocity based on the overall velocity and direction of all nodes in a group, with each node having a deviation from this mean which represents a node's real velocity and direction. By modelling node velocities, nodes join groups with velocities similar to theirs, regardless of physical location. This is a very different approach to more traditional group mobility

models, but demonstrates the more unusual approaches designed to improve connectivity using past and current mobility data alone. By using signal loss maps and real future node mobility predictions, the resulting algorithms could be far more accurate.

An alternative approach to predicting changes to network topology has been developed by Goff, Abu-Ghazaleg, Phatak, and Kahvecioglu [GAP01] in the form of Preemptive Routing. The basis of Preemptive Routing is a mix of on-demand and table-driven routing protocols. Using on-demand routing minimises overhead in general, as only the required routes are broadcast, not entire routing tables broadcast periodically as is the case of traditional distance-vector routing protocols. However, table-driven routing protocols have the advantage of constantly updating routes, so that unstable routes, using a suitable metric, are not used in favour of higher quality routes. In [GAP01], routing is based on on-demand algorithms. However, path discovery is not only initiated when a node is looking for a path, but is initiated whenever a path is likely to break. The algorithm pre-emptively switches to a "higher quality" path when it becomes aware of a potentially failing path. It accomplishes this by monitoring the signal strength of links, reinitiating route discovery on learning that a link has reached a certain signal strength, predefined as a "Preemptive Region". Once a link's signal strength enters this region, the route discovery process begins and the best alternate path is selected.

The authors note that several other criteria may be used for the pre-emptive warning, such as the age of a path, number of hops or rate of collision. They also note problems with such an algorithm, such as unnecessary overhead if a node going outside communication range changes direction and comes back, making a path change unnecessary. Having specific knowledge of where nodes will travel, such as the situation of this thesis, creates new possibilities and greater optimisations to overcome some of these issues.

2.2 Signal Loss Maps

Several authors have considered the idea of creating a *signal loss map* detailing a logical topology of expected signal capability. Such signal loss maps, also known as signal strength maps or radio frequency maps, detail how radio waves propagate over various physical areas. Fritsch, Tutschku, and Leibnitz [FTL95] identified two key components of signal loss maps, that they are both location-varying and time-varying. They propose

two map generation strategies for generating data, though do not focus on the map itself. One strategy is called the Line-of-Sight approach, where the signal loss is calculated by taking the shortest, direct path between transmitter and receiver, while also considering any obstructions along this line as increasing the signal loss. Their second approach is based on ray tracing where, as in 3D graphics, rays are traced from the source (the transmitting node) in all directions, taking into consideration the reflection and diffraction on various objects in the scene, until the receiving node has received the signals. This method, as they emphasise, is computationally expensive.

Hassan-Ali and Pahlavan [HP02] have also looked at using a physical map to predict signal loss, and use the ray tracing approach, except instead of using a very detailed map, they speed up ray tracing by counting the number of obstacles and applying a combination of heuristic formulas on this information to generate a prediction. Both approaches are based on the idea of a detailed physical map of the environment, which has the potential to produce highly accurate results depending on the detail of the map. However their approaches cannot provide a useful solution if no detailed physical map is available. In several present real-world scenarios (e.g., disaster recovery sites and urban environments where the physical environment may be constantly changing due to traffic for example) this is more likely the case.

Existing solutions for real-world scenarios focus on building the signal loss map prior to the map being used, and also focus on creating the map using fixed beacon nodes, not from autonomous neighbouring nodes, as applies to ad hoc wireless networks. Howard, Siddiqi, and Sukatme [HSS03] propose a signal loss mapping solution for use in mobile robot localization. In their solution, a signal loss map is built from samples collected between a mobile node and several fixed beacons. Using a low-pass filter on the samples collected, a grid of fixed-size cells is generated. Given a static environment with fixed-position beacons, this signal strength map produces good results. Their work is a significant contribution to the field because information is collected entirely from wireless devices; there is no need for a physical map in order to build the logical signal map. However, the solution explored in [HSS03] is based on localization, not on providing a detailed logical view of signal strength capabilities within an area. Their work is concerned with identifying a node's location based on the relationship of signal strength between each of the beacons. Because of this, their goals leave open areas that

perhaps can be further explored for use in continually-evolving signal loss maps for use in wireless ad hoc networks. Their signal strength map has several other issues which have not been addressed and are worth exploring. These issues include:

1. Collection and collation of data is done only once. In an automated network, where map knowledge may continually expand and signal strength may vary in an area over time, a solution must continue to update the map in real time.
2. Because no data is collected and collated in real time, no concept of weighting more recent data with more importance has been considered.
3. The map proposed in [HSS03] has a fixed resolution. While the size of each cell can be fixed at any size, in a realistic environment there may exist large areas with the same signal capabilities and others that require much more detail. Depending on the environment, a map should be able to vary the size of its cells automatically.
4. Each map only deals with one destination node, where many nodes can record their signal strength to only one single fixed beacon node per map. The map thus has a M:1 relationship, and does not detail the capabilities of a cell to any other cell.
5. Because of the aforementioned reason, no concept of direction of signal is introduced, where a signal from one cell travelling in one direction may propagate differently to a signal travelling in another direction.

[HSS03] presents one of the most significant contributions to the relatively unexplored field of signal loss maps for use in mobile wireless networks, but leaves room for many improvements where ad hoc networking is concerned.

Youssef, Agrawala, and Shankar [YAS03] have considered problems with some traditional signal loss maps as part of their location determination goals. They identify a common issue where traditional signal loss maps have two phases, an offline phase to develop the map and an online phase where the map is used. In these systems, signal data is collected to generate maps before they are used, making the maps static (such as with [HSS03]). In [YAS03], the offline phase still exists, and still generates the core signal loss map. However, during the so-called online phase, any variations to signal loss over time are attributed to noise characteristics, and a scheme is derived to handle them for their

purpose of location determination. They overcome noise issues by requiring several fixed infrastructure nodes to be within range of each node, and so an average from each beacon forms the basis for determining the location of possible noise, and thus the adjustments which should be made. As with [HSS03], the focus of [YAS03]'s work has been on improving location determination algorithms for mobile nodes, and thus has the same issues when applying such algorithms to wireless ad hoc networks.

2.3 Routing

Ad hoc networks depend on connectivity between nodes, and many routing protocols have been created for wireless ad hoc networks to date. The effectiveness of many of these routing protocols could be improved if they had knowledge of future node connectivity.

One of the first ad hoc routing protocols was Destination-Sequenced Distance Vector (DSDV) [PB94], a single-hop-based distance vector routing protocol. In DSDV, each node has a routing table which contains the details of the next hop for each destination it can reach, along with the number of hops to that destination and a sequence number. The sequence numbers are used to distinguish stale routes from new ones, thus avoiding the formation of routing loops. When a destination node sends out a broadcast, it assigns it a unique sequence number higher than the last, thus ensuring that receivers know this is new information from the destination. Any nodes receiving an update will use the highest sequence number. If two updates have the same sequence number, the route with the smaller cost metric will be used. DSDV alleviates table update flooding by employing two types of update packets. A "full dump" packet carries all available routing information, used when network changes are high, when consistency checking is required, and when new nodes join the network. Smaller "incremental" packets are used to relay information that has changed since the last "full dump".

Aside from the sequence numbers, DSDV follows the common table design of generic distance vector routing protocols. Implementing two different broadcasts of routing information, "full dump" and "incremental", cuts down the amount of traffic generated in traditional routing algorithms, which is vital as packet collisions are more likely with a wireless network, and when battery power becomes a factor. DSDV successfully introduces distance vector routing to the wireless ad hoc environment.

Dynamic Source Routing (DSR) [JM96] is an on-demand source-initiated routing protocol. It is one of the first designed for ad hoc wireless networks and one of the simplest protocols to understand. Each node is responsible for generating a route to any destination node through the route discovery process if there does not already exist a route in its route cache. If a route exists in the routing cache with a valid unexpired timeout, that route is used to send any waiting packets. If not, the source node broadcasts a route request to nearby neighbours, containing the source and destination address, an identification number and a route record of the nodes that pass on this packet and do not have a route yet to the destination. When the destination or some other node with a valid route to the destination is reached a route reply is sent back with all nodes along the path to the destination in the route record (if an intermediate node replies, then it will append the route record from its cache to the current route record). To avoid loops in DSR, a mobile node will not forward a packet where its address is in the route record. Route maintenance is performed regularly to clear the cache of invalid routes. Routes are removed from cache after the node along a path fails, if an acknowledgment fails to get through, and if this node passively hears similar failure circumstances from neighbouring nodes.

DSR is a simple protocol that acts similarly to link-state algorithms in the way it can build up information about the network. With an appropriate route-weight metric, it would be possible for a node to make use of multiple paths to a destination, and make a well-informed decision as to which it should take. The only disadvantage to DSR in a mobile environment is the significant increase in bandwidth when a route is being searched, as the entire network is flooded with requests to find paths to the destination.

Temporally-Ordered Routing Algorithm (TORA) [PC97] is a distributed routing protocol that aims to reduce the communication overhead commonly associated in other on-demand protocols, such as DSR. When a node requires a route to a destination, it broadcasts a query packet containing the address of the destination required. This packet propagates through the network until it reaches the destination or a node with a route to that destination. This node then responds back with an "update" packet listing its "height" in regards to the destination node, which is essentially the distance from that node to the destination. Each node passing the "update" packet along then increments

the height accordingly, and records the height, destination and next hop node in its own cache. This creates a list of routes in order to get from the original sender to the node that initially generated the "update". As these packets do not contain every node from the source to the destination, overhead is significantly less than DSR. In TORA, only the next hop is ever remembered. It differs from DSR mainly by localising control messages to its nearest (1-hop) neighbours only, reducing the communication overhead of broadcasting a list of all nodes from the source to the current node searching for the destination.

The use of node location to aid routing has been researched previously. Most location-based routing algorithms to date have been centred on conserving energy or reducing bandwidth associated with routing. The specific aims of this thesis are in ensuring connectivity through the use of future predicted location information, although some preliminary concepts can be gained from existing location-based routing protocols.

Location information is used to limit the broadcast range and direction of packets as they flow through a network. While the direction of wireless signals cannot be altered by software algorithms alone, it is possible to select which intermediary nodes should forward packets to a destination based on their location. Using location information to aid route discovery often works best with on-demand routing strategies (such as DSR and TORA) as other routing strategies tend to receive route information, rather than search out for it from the source node's point of view. One of the first examples of such a protocol is in the Location-Aided Routing protocol (LAR) [KV98]. In LAR, a source node attempting to find a route to a destination node can limit the area of potential nodes having a route to the destination node by using location information of both itself and the destination node. An area covering these two locations forms a "request zone", which limits the extent to which messages are flooded in order to find a route to the destination, reducing traffic in the network by localising such broadcasts to nodes only within the request zone. This concept helps reduce traffic, but leaves unaddressed the issues of increasing connectivity using such location information.

2.4 Autonomous Robotics

The study of autonomous robotics provides some fundamental concepts on which predicted routing is often based. Mobility patterns and control of autonomous nodes are important to predicted routing algorithms in determining future node positions. Future node positions form a stable basis for prediction when combined with a signal loss map.

Predicted routing benefits greatest where future node mobility is known and structured. The travel paths of autonomous nodes are variable and do not often follow straight lines. This is due to obstructions in the intermediate environment which require collision-avoidance [Ono00]. However, the movement of autonomous nodes is, in general, structured. Several strategies (e.g. [ABO05], [FT02], [OC96], and [THM95]) use checkpoints to mark locations where a node will travel. Although the path between the checkpoints is unknown, it will be as close to a direct path as possible given the terrain and obstacles that lie between the node's current and intended location. This information is usable by predicted routing algorithms to improve connectivity in wireless networks.

[ABO05] uses Milestone Positions (MPs) to mark checkpoints along a path so that a robot can return to its initial "home" position. Collision avoidance is used between MPs (using a system such as [Ono00]), but minor collision avoidance is irrelevant where the entire general path is considered. While [ABO05] studies only a specific robotic function of returning home, the overall design of checkpoints can be applied to many robotic applications. [ABO05] tested the use of MPs to allow a robot to return to its home position on a robotic platform in an indoor office environment. The tested robot was able to achieve high accuracy after long journeys during which the robot performed many complex pathing manoeuvres using MPs.

[FT02] creates a path of waypoints to a destination by finding points which are likely to be free of obstacles. These "hot points" are defined as areas which a robot will aim for, defined by:

- how far the hot point is from the goal position,
- whether the hot point is occupied by a static obstacle,
- whether the hot point is occupied by a moving obstacle, and
- whether the hot point will be occupied and how soon by a moving obstacle.

[FT02] uses a neural network for short-term (one step ahead) and long-term (many steps ahead) prediction to change a robot's route towards its destination position if there will be obstacles in the current path. The new path is formed by hot points as they are discovered and required. Both [ABO05] and [FT02] are examples of using checkpoints to mark future node positions, which are usable by predicted routing algorithms.

2.5 Deficiencies in Current Research

A limited amount of research has been conducted into using future node mobility predictions to improve connectivity in wireless ad hoc routing protocols. Existing routing protocols that use prediction have focused on current node mobility and network conditions as a means of prediction. [WL02], for example, categorizes nodes into mobility groups. A node is thus predicted to travel along with its group, wherever that group may travel. Other predicted routing protocols, such as [GAP01] and [CJS03], do not rely on any mobility pattern, but instead monitor the signal quality between nodes and predict disconnections based on a weakening signal. These protocols do not consider predicting the unique future movement of each node. [Su00] took into consideration each node's unique bearing and speed of travel, but failed to consider any path beyond this, where entire journeys have predicted node movement. The emphasis of future movement is further highlighted where routing protocols are applied specifically to automated networks.

Existing predicted routing protocols have assumed a free-space signal loss model for prediction. Work by [FTL95], [HP02], [YAS03] and [HSS03] have all identified the need for signal loss maps where knowledge of the wireless propagation environment is necessary. However, these signal loss map solutions are poorly suited for use in wireless ad hoc networks, as they are unable to develop in real time using only wireless nodes. In ad hoc networks, a wireless infrastructure and existing signal loss maps of the environment cannot be relied on. Nodes are required to construct signal loss maps themselves, and be capable of adapting to changing environmental conditions (such as weather cycles). The need for a signal loss mapping solution specifically designed for wireless ad hoc networks has yet to be achieved.

Routing protocols to date have been focused on improving connectivity by improving recovery time of disconnected routes, while overall aiming to reduce bandwidth cost of

routing protocols. The use of predicted connectivity in wireless routing allows potentially all disconnections to be pre-empted and alternate routes to be discovered ahead of time. Specifically in automated networks, the predictable nature of node mobility can be used to improve existing routing protocols.

2.6 Summary

Existing research has presented various approaches to the three components of prediction in wireless ad hoc networks: predicting connectivity, signal loss maps and routing algorithms. Only a few of the vast number of routing protocols have been presented in this research, as prediction is capable of being applied to any routing protocol that uses a metric to select routes. Previous studies into using prediction and in signal loss maps have been limited. It is the purpose of this thesis to further investigate prediction for use in wireless ad hoc networks, as will be detailed in the following chapter.

Chapter 3: Objectives and Framework

3.1 Objectives

The review of current research into prediction in ad hoc networks highlights many issues that need to be explored. The aim of this research is to explore the area of prediction in ad hoc wireless networks to identify whether prediction can improve reliability of routing. In order to achieve this, this thesis presents and tests a novel set of algorithms to increase the stability of routes in wireless ad hoc networks. This is accomplished through the use of predicted node movement information combined with detailed information of the wireless signal loss environment. This thesis aims to provide predicted information in a form usable by any routing protocol employing a metric for route selection. This thesis will present a method to provide predicted route connectivity, based on future node movement and a signal loss map. A signal loss map capable of being developed and updated in real time by mobile ad hoc nodes is also presented.

3.2 Framework

This dissertation has been divided into several sections to facilitate the explanation of the design and implementation. The end result of the research is a set of algorithms capable of providing prediction to wireless routing protocols.

No prediction of connectivity between wireless nodes should be attempted without incorporating some form of signal strength measurement. Physical environments all have different features, atmosphere and structure which greatly vary how a wireless signal will propagate towards other nodes. To predict two nodes having connectivity requires the prediction of not only their future location, but also the signal loss between these two future locations. If future signal loss predictions are made by using previously recorded signal losses for the same locations then it is possible to reuse this information. However, it is nearly impossible to record signal losses between every two locations in a physical

environment. The number of positions from a combination of two arbitrary node locations is infinite. As such, it is suggested that signal loss recordings can only be used to create a map of average signal loss propagation over various areas of a physical topology. In order to accurately use predicted locations, a wireless signal topology map is required. Such a logical map defines how wireless signals lose strength as they travel through physical space. Such a map is a significant challenge to create, as there is no tool or equipment which can currently provide the three-dimensional signal propagation model of an environment. Even if there were, the unknown environments that nodes will enter and the prediction of the environment in the future may still represent a challenge. The Communication Map presented in this thesis is a signal loss averaging map, collected and combined by all nodes in the network. Signal losses between nodes are collected and averaged to form a logical map of how signals propagate over various physical locations. Combined with the predicted future physical locations of nodes, this forms a basis for predicting connectivity of nodes in a network.

Node mobility in this thesis is being provided by a simple task list. This list contains the times and locations where a node will travel. How this task list is generated is outside the scope of this work. However, a common means of creation would be by autonomous robots, which will likely have such a task list already implemented in their own algorithms. The task lists of autonomous nodes represent where they will go and when, where autonomous nodes may be sharing such information with other autonomous robots in a group.

In order to improve ad hoc routing algorithms using prediction, a layer of abstraction is provided to separate prediction algorithms from routing algorithms. This top layer is the Future Neighbours Table, a listing for each node of when all other nodes in the network will have or not have connectivity with the listing's node. The table provides immediately available and continually updated information on the future topology of the network from the point of view of each node. Each node, in having a list of predicted neighbours and the periods they will be neighbours for, can incorporate that information with ease into existing ad hoc routing protocols. For example, in selecting the best path where two or more paths have the same hop count, the path which will remain connected the longest could be selected. It is trivial to calculate this given the layer of abstraction provided by the Future Neighbours Table. The Future Neighbours Table itself calculates

the predicted future neighbours based on information about nodes' future mobility, combined with a map of the wireless signal topology. The Future Neighbours Table uses the Communication Map to obtain an estimate for the signal loss between any two physical locations.

Though the Future Neighbours Table can be applied to any number of existing routing protocols, both a link-state and a distance vector algorithm are tested which make use of the predicted information algorithms presented in this work. They are compared with a number of other ad hoc routing protocols to analyse connectivity improvement.

This thesis contributes primarily four components to the field of wireless ad hoc networks. Firstly, the Communication Map (CM) is the signal loss mapping solution presented in this work, and is created from wireless signal and location information from wireless nodes. Secondly, the Future Neighbours Table (FNT) uses this information in combination with future node movement information to provide existing routing protocols with connectivity predictions. The relationship between these components is highlighted in Figure 1. Thirdly, as existing simulator tools are not suitable for evaluating the performance of the CM and the FNT, a custom simulator tool is created. Finally, significant numerical results of applying the CM and the FNT to existing routing protocols are studied. Following this chapter, Chapter 4 will present the architecture and algorithms for the CM. Chapter 5 will go on to use the CM in the algorithms to create the FNT. Chapter 6 will apply these concepts to existing routing protocols. Chapter 7 will present the custom simulator created in this thesis, while Chapter 8 will study the performance of the CM and the FNT on existing routing protocols.

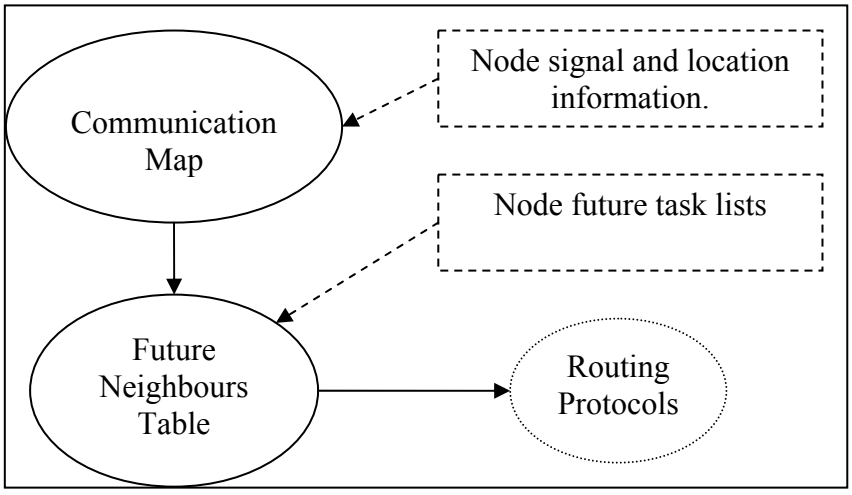

Figure 1: Framework

Chapter 4: Signal Loss Maps and the Communication Map

4.1 Motivation

In any wireless networking environment where nodes use predicted location information in order to improve routing, the ability to predict signal communication strength from one location to another is vital. However, a prediction on the wireless connectivity between two nodes cannot rely on location information alone. The majority of contemporary wireless networking environments feature many obstructions which reflect and block wireless signals. Consequently it cannot be assumed that all signals between two physical locations will travel with free-space signal loss, such as is relied upon in [Su00]. In addition to requiring knowledge of nodes' physical locations, the predicted propagation of signals over physical areas is required. Wireless transmission capabilities in an unknown or known environment are, however, nearly impossible to predict with perfect accuracy [HSS03]. Not only do different locations have different propagation characteristics, but environmental conditions may change those characteristics over time. Foreign nodes operating on the same channel, radio-frequency interference, environmental noise and even landscape may change radically over time, affecting any recorded or estimated measurements of signal loss.

To overcome these challenges, a signal loss map is required. Signal loss maps represent the logical signal propagation topology over a physical area. They describe how signals are likely to propagate in various directions over various distances. Due to the constantly changing nature of the wireless environment, a perfect signal loss map is not possible to create with current technologies. However, various estimates may be developed to provide, with appropriate safety margins, predictions on whether two nodes at different locations will have connectivity in the future. The objective of this chapter is to present a novel signal loss map solution to overcome these challenges. This solution provides the Communication Map (CM) component of the framework described in Chapter 3.

4.2 Issues that are raised

Generating a signal loss map is challenging for many reasons. The challenge in creating a wireless signal loss map is to represent the propagation of signals over the area used by nodes in a wireless ad hoc network. This map can then be used to predict the future signal loss between any two arbitrary positions in a manner that can be shared among nodes with reasonable accuracy yet minimal size and lookup time. The issues that challenge this research are important to consider.

The signal propagation over various areas is likely to change over time due to any number of external environmental conditions. Such outside factors will always lead to unpredictable interferences in unknown environments. There are issues of other wireless-transmitting devices in the environment, day/night cycles and their effects, weather and storm conditions, etc. All of these factors can potentially influence the propagation of signals significantly. It becomes necessary to move away from the one-time sampling from beacon nodes that several algorithms employ (such as in [HSS03] and [YAS03]) and instead monitor the signal propagation characteristics over the entire map continuously. To meet the requirements of this thesis the process must also be completely automated. Therefore some form of averaging of signal loss measurements over time, focusing more on recent information (a weighted average), is a vital part of updating a signal loss map with new signal strength information.

Another important issue is the lack of a known signal propagation range. The strength of a transmission from one location can be dramatically different to a position nearby, due to environmental landscape factors (such as walls). It is inefficient to have every square inch of a bounded environment mapped with its signal strength, both in terms of collecting the information and in storing and sharing it with other nodes. It even becomes impossible if the signal strengths from one location to another may change over time. It is infeasible to collect samples from all locations at all times without a grid of wireless nodes, which is not a feature of ad hoc wireless networks.

Further to this problem, a single node cannot cover as many locations as many nodes can. Some nodes may be stationary for long periods of time, and never collect the samples needed to cover map sections that they might require for future predictions. A set of nodes are more likely to generate a more detailed and up-to-date map as a whole, rather

than each node individually. Therefore it would be beneficial to provide a means for nodes to share their findings with neighbouring nodes, while keeping bandwidth overhead to a minimum. With possibly hundreds or thousands of signal measurements taken every second, this information itself cannot be efficiently propagated to all other nodes. Rather, some mechanism of building and sharing a condensed node's map is required.

There is also a need for the signal loss map to represent the logical signal topology while remaining relatively compact. Current wireless ad hoc routing protocols are used on a variety of platforms, from PC-based robots to small mobile phones, where the storage capacity may be very limited. Coupled with the requirement to share maps, this presents a challenge to represent a large area of differing signal propagation from a small map.

In addition to maps being employed in a variety of environments, different areas of an environment may require varying levels of detail. For example, in street maps cities and city centres often have a much greater level of detail than country roads and highways. Likewise with signal loss maps, there will be some areas where signal propagation is largely constant over a very large physical area, such as a football field. There may be other areas where a much higher level of detail is required, such as inside an office building. An ideal signal loss map will be able to cater for various levels of detail as they are uncovered.

The issue of unmapped terrain (in terms of recording signal loss averages) is also of importance, as nodes may have to travel to unmapped locations. These locations may initially be unmapped, as the network develops the signal loss map in real time. This is not particularly useful if future predictions are required for areas of the map that are unmapped. If a node is heading to an unmapped signal location, an estimation of communication capability will be required given any surrounding information, including environmental information. The ability to predict with some accuracy the transmission capability of an unmapped location is perhaps one of the more challenging tasks of generating a signal loss map.

Finally, signal loss maps in ad hoc networks have only signal loss measurements per packet as a means of determining signal propagation. In constructing a view of the

wireless signal propagation environment, it is impossible to accurately map signal loss over physical areas with such limited input. The ability to derive signal propagation from node position information and signal loss measurements is the most significant challenge to real-time signal loss maps in wireless ad hoc networks.

In summary, the issues concerning the use of a signal loss map include:

1. Communication capabilities will change over time.
2. Signal propagation will vary from different locations.
3. Several nodes generating a shared map will be better than a several nodes each creating individual maps.
4. Signal propagation areas of detail may be of varying sizes.
5. Signal propagation may be needed for unmapped locations.
6. Signal loss map size for sharing and storing should be kept minimal.
7. Signal loss readings must be mapped appropriately over physical areas, even though there exists no perfect means of doing so.

The above issues are a requirement for any signal loss mapping solution for use in autonomous wireless ad hoc networks. This chapter presents a novel approach to representing signal propagation over a physical space. The solution presented is designed specifically for use in prediction algorithms, although it can be applied to a wide variety of signal-mapping requirements. This chapter is organised as follows: the design of the proposed solution is presented, followed by algorithms to develop it. Finally, an alternate solution is proposed which improves the original solution.

4.3 Map Design

A signal loss map is designed in this thesis to overcome the challenges previously described. This signal loss map, the Communication Map (CM), is tailored to be built in real time using only wireless ad hoc nodes. The map is created using signal strength information provided with each packet as it is received from any node's wireless interface hardware. To provide a physical reference system, some form of location-providing device is required for each node. In this research, a system such as GPS [EM99] is assumed to be available to provide the coordinates of each node. From these two external sources the CM is constructed.

As stated previously as Issue 3, one of the challenges of signal loss maps is that they be created with the assistance of all nodes in the network. This raises the question of whether the CM should be centralised or distributed in its design. There are benefits to a single shared map approach, primarily in providing protocols relying on CM information with a consistent view of signal topology among nodes. A single map aids node calculations concerning that map where different nodes may require the same conclusion. For example, two neighbouring nodes may both need to agree that they are neighbours in order to communicate at some future point in time. However, due to the constantly-changing nature of the map and the potential for the rate of propagation of the map to be delayed significantly, a single common shared map cannot be achieved. The CM is therefore node-based, where each node has its own representation of the wireless topology. These maps are heavily shared and merged together as a large form of input in creating each node's individual CM. However, this map is formed unique to each node.

Signal strength loss, not signal strength, is used so that a predicted signal passing over multiple areas between two nodes can be calculated and its predicted signal loss can be removed from the acceptable signal loss between two nodes. This can then be used to calculate the expected distance a signal will travel, which in turn can be used to decide whether two nodes at future locations will be able to communicate (as neighbours) or not. The maximum Acceptable Signal Loss (ASL) between two nodes is the sum of a node's output power with its receiver sensitivity [Sha01]. For simplicity, it will be assumed that all mobile nodes in a network using this map will have identical hardware; though with some modifications it would be possible to extend the functionality of this signal loss map beyond this requirement.

4.3.1 Representing areas

The proposed solution in this research follows a similar approach to that of Howard et al. [HSS03], exploring how areas having a certain average signal strength propagation characteristic can be represented. To represent areas, shapes are used to create a boundary around some logical area where the signal loss through that area is mostly consistent. These areas summarise data collected in such areas, both from a node's own signal loss calculations and the sharing of data from other nodes as sources. In summarising this data the memory footprint of the signal loss map is kept low, which has been identified as Issue 6 of the challenges of signal loss maps. A reduced map size helps

both low-memory devices and in sharing maps over low-bandwidth wireless connections. However, this can have deleterious effects, which are discussed later.

A 2D map is preferred for the purposes of describing and testing algorithms, leaving the conversion to a more realistic and complex 3D model for future work or implementation. In using a 2D implementation, the aim is to validate concepts and provide a simple overview of operation. With either approach, the basic shape design remains the same. The most accurate, but hardest to work with, is a free-form design, with the number of sides and the overall shape continually evolving depending on the information being represented. This model requires substantially more algorithmic calculations to be performed in both creating a map and in sharing it with other devices, in contrast to Issue 6. As this research is not focused on the physical mapping of wireless networks but rather using them in predicting node connectivity, such detailed representations are ill-suited. The purpose of the CM in this research is to represent the estimated communication capability between any two nodes on the map at any future point in time. As the information in this map is used regularly by routing and the Future Neighbours algorithms, the emphasis for the CM is in calculating this information quickly.

To this end, a shape is required that will not change its constitution nor require overlapping to cover an entire map. Both of these would add complexity which is preferred to be avoided. The primary candidates are a square, a triangle, and a hexagon. The easiest shape to operate on and share is a square. A square is applied easily to define a map of non-overlapping areas. This concept of a square shape to represent areas in a CM will be used in this thesis; however it is identified as an imperfect concept. There is room for future work, and to aid in this, the map will be designed around the concept of cell objects, which package any particular shape or form that will represent areas on the map. The only condition put upon extensions of a cell is that the concept will centre around non-overlapping shapes. To go into the depth of overlapping shapes is beyond the work of this thesis (see Section 4.6). A cell represents the signal capabilities of an area in the environment through which an automated network would be operating. Each cell represents an area where the signal propagation characteristics are considered to be uniform, and thus a future signal being transmitted from this area would be expected to behave similarly. Figure 2 illustrates a signal being transmitted over an example map of cells.

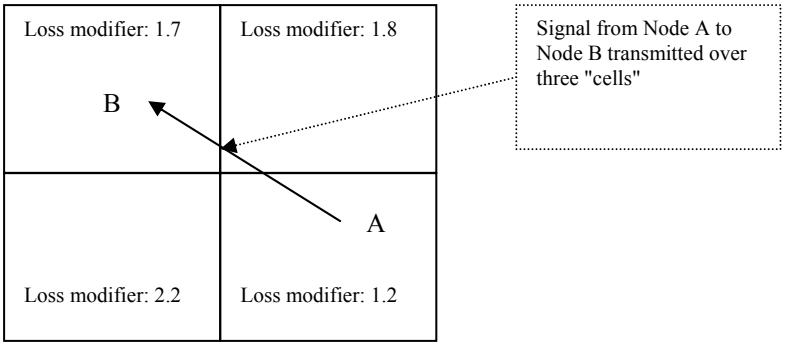

Figure 2: Signal passing through multiple cells

4.3.2 Continually Evolving Map

The map can expand in any direction as new areas are discovered. The use of squares for this map prevents overlapping if equal-sized squares are always created initially, starting with some optimal sized square. This optimal sized square (Default Cell Size, DCS) is selected as a square which encompasses the maximum signal travel distance of a signal from a specific wireless interface in perfect conditions of a free space environment. For the benefit of this research it is assumed that signals will not be naturally amplified by the environment, even though in rare cases this is possible [AB25]. This defines a cell with a maximum size and thus optimises the size of the map by minimising the number of unique shapes, lowering memory usage to its minimum (Issue 6).

The actual environment might require a more detailed map, depending on specific instances (Issue 4). To address this issue, the concept of subdivisions is introduced. In this approach, a cell will be subdivided into smaller cells if the signal loss between these smaller "sub" cells is greater than some minimum acceptable value (Minimum Cell Subdivide Difference, MCSD). Equally, if in time it is discovered that a set of subdivided cells no longer hold a signal loss significantly different from each other (lower than some minimum merge difference) the cells can be merged back into the original cell. This value must be lower than that of the MCSD, so that cells do not divide and merge inconsistently (defined as Minimum Cell Merge Difference, MCMD). In the case of square shapes for cells, a square is subdivided into 4 equal parts should the signal loss be significantly different internally within the cell.

To accomplish subdivisions, each cell must already be recording subdivided data. This is to know if different subdivisions have a significant difference in signal loss, but are as of

yet not using such data for predictions. These subdivisions will be known as *potential subdivisions*, becoming subdivisions if the MCSD is reached. Whether the subdivisions are potential or actual, recording data for them is the same. Each cell updates its own average (discussed below), and calls upon each of its subdivided cells (if it has any) to update themselves too. A *potential cell* does not have subdivisions, as there would be an infinite level of detail recorded on a map, defeating the purpose of the optimum cell size concept of keeping the memory footprint of a map as small as possible. However, *actual cells* should have *potential subdivisions*, and cells which are already subdivided do not have four *potential subdivisions* at all, but rather four *actual subdivisions*. These cells will be known as *parent cells*, cells that are already subdivided and contain child cells.

As an example of the subdivision and merge variables, consider Figure 3 and Figure 4. Figure 3 shows an *actual cell* with four *potential subdivisions*. If the MCSD is 2.0, then the *actual cell* would subdivide, as the difference between the highest (3.5) and lowest (1.2) modifiers is greater than 2.0. Figure 4 shows a *parent cell* with four *actual cells* as subdivisions. If the MCMD is 1.0, then the *parent cell* would merge the subdivided cells back together, as the difference between the highest (2.0) and lowest (1.2) modifiers is less than 1.0.

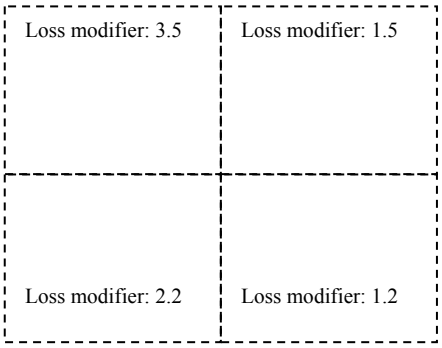

Figure 3: Actual Cell with Potential Subdivisions

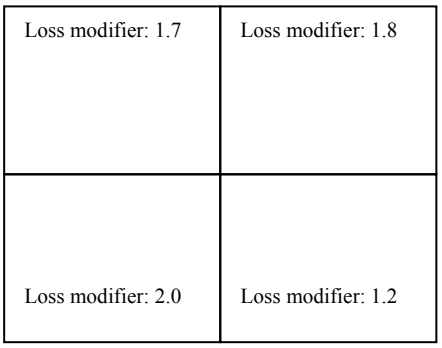

Figure 4: Parent Cell with Actual Subdivisions

Each cell on the map can therefore be in one of three states:

- *Parent cell*: A cell which has *actual subdivisions*. This cell will use the subdivisions' average signal loss for a more accurate prediction of its signal loss and not its own.
- *Actual cell*: A cell which is being used on the map. This cell has its own signal loss, but also has a set of subdivided *potential cells* to monitor the need to subdivide, where it would thus promote itself and become a *parent cell*.
- *Potential cell*: A cell which records signal loss, but the signal loss data is not currently active and is not used in predictions. This cell has no subdivisions, and is used simply to monitor the need for its parent *actual cell* to subdivide.

The decisions on subdividing or merging come after the updating process, when all cells and their subdivisions have been updated from an incoming signal, and the cells' average signal loss value may have changed. It is at this point that the decision as to whether to subdivide a cell (if it is an *actual cell*), or reunite subdivided cells (if it is a *parent cell*) is made. If a cell is an *actual cell*, the process is trivial. An *actual cell* will have *potential cells* as subdivisions. If the difference between the highest and lowest value subdivided cells is greater than the MCSD, then the current cell will become a *parent cell*, with each of its subdivisions being informed to become *actual cells* (at which point in time they will create *potential subdivisions*, allowing the process to continue). This process can continue until the Minimum Cell Size (MCS) is reached, preventing detail from becoming too fine. If a cell is a *parent cell*, it will only consider merging if the difference between the highest and lowest value subdivided cells is less than the MCMD. Additionally, a *parent cell* may only merge subdivided cells if all those cells are *actual cells*. If one or more subdivisions is a *parent cell*, even if the average of this *parent cell* is similar to the other subdivisions and would normally warrant being merged, being a *parent cell* signifies that a greater level of detail is required at some lower level. Because of this it is important that all subdivided cells that are not *potential cells* (as they have no subdivisions themselves) evaluate the status of their subdivisions first, before *parent cells* do. This algorithm to evaluate subdivisions is described in Algorithm 4.1.

Algorithm 4.1: Evaluate Subdivisions

- Let *h* represent the highest signal loss.
- Let *l* represent the lowest signal loss.
- Let S_{loss} represent a subdivided cell's average signal loss.

1. Iterate over each subdivision, finding the highest signal loss (h) and lowest signal loss (l) of all active subdivisions:
   a. If the subdivision is not a *potential cell*, repeat the *evaluate subdivisions* algorithm for the subdivided cell.
   b. If the subdivided cell's S_{loss} value is higher than h , let $h = S_{loss}$.
   c. If S_{loss} is lower than l, let $l = S_{loss}$.
2. If the cell type is an *actual cell* and the difference between *h* and *l* is greater than *MCSD*, and the *MCS* hasn't been reached, convert the cell into a *parent cell*.
3. Otherwise, if the cell type is a *parent cell* and all subdivisions are *actual cells* and if the difference between h and l is less than *MCMD*, convert the cell to an *actual cell*.

Algorithm 4.1 is executed on every *actual* and *parent cell*. In step (1), *h* and *l* are found from all active subdivisions. A subdivision is active only if it has received signals or obtained averages from another node's CM during its history. This prevents subdivisions from occurring where a cell would subdivide simply because three of the four potential subdivisions had never received a signal, thus are left with using the default modifier of 1.0 (i.e. a wireless signal travelling in free space without any interference). In step (2), a cell is subdivided if the MCSD is reached, so long as the cell analysing its subdivisions does not have a size equal to the *minimum cell size* (MCS). The MCS prevents a CM from describing an infinite level of detail. If cells are able to subdivide down to the smallest unit of measure, not only will bandwidth usage increase dramatically when sharing of CMs occur, but the purpose of summarising areas of similar signal propagation will be defeated.

4.3.3 Example Map

This approach of using cells describes to users of the CM the same information that vendors of wireless cards use to describe range and signal strength capabilities. Vendors of wireless cards often include the maximum range and signal strength of their product in a variety of general scenarios, such as outdoors, home environment and cluttered office. Similarly, the CM generates such scenarios in real time and delineates where on a map such areas exist.

An example of this approach can be seen in Figure 5. In this example there is an outdoor area, with a building on the right and a small forest of a few trees on the left. All areas were originally divided into equal squares, where the squares are subdivided further if the signal loss is significantly different inside a cell. The dotted line shows where a cell has been split into four equal parts by the CM, but where a human would read the figure as rectangle areas. Each cell contains a *modifier* value, which represents the increased rate of signal loss. The *modifier* value is detailed later in Section 4.3.6. The work of the CM in this thesis is to create and represent such a map that can adapt to an unknown environment and represent the signal propagation over its various areas.

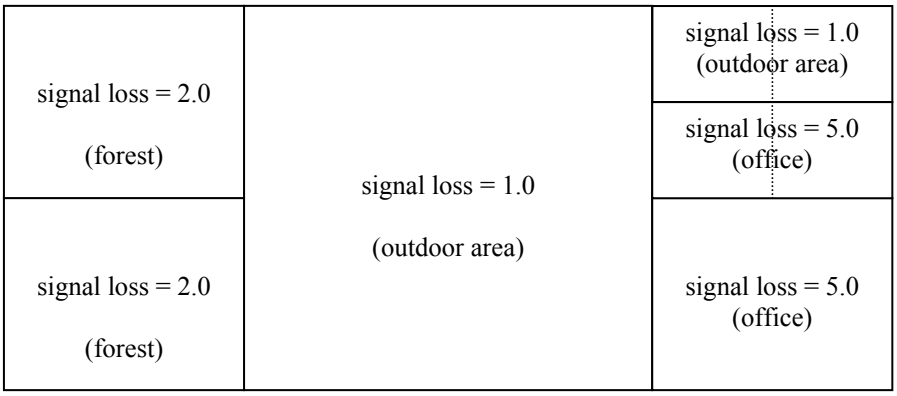

Figure 5: Example CM

4.3.4 Time

The function of a cell is to average signal loss within its bounds. In wireless environments, signal loss can vary greatly over time. In most environments one could, however, expect some degree of uniformity to appear. This average may change over time as environmental conditions vary. In recording an average, it is the most recent data which has importance, where previous data collected is gradually made redundant over time. The collection of all signals in an offline phase combined with a low-pass filter to provide future estimations is used in [HSS03]. This approach could be extended to collect signals in an online phase, discarding signals once they have aged some period of time. However, one of the goals of the CM is to minimise its size, and storing every signal recording is contrary to this aim.

There are several difficulties in applying traditional windowing techniques [OS89][LF98][MF53] with signal collection for the CM. Firstly, the CM is intended for use on low-capability devices. Limits to processing power and memory availability affect

the amount of data that can be stored and used for a signal loss map solution. Traditional methods rely on sampling and often time-stamping individual signal recordings, which leads to a potentially large collection of samples. Secondly, these samples also need to be shared and broadcast to neighbouring nodes if other nodes are to rely on the same information. Thirdly, sampling data periodically presents additional problems when all nodes may not be transmitting periodically, as nodes need only transmit packets when required. Any approach where individual signal readings are collected and stored is considered infeasible due to the potential quantity of signals and the requirement of map sharing between nodes. As such, window averaging algorithms such as Hamming window, triangular windows [OS89][LF98] and Fourier transformations [MF53] cannot be applied directly to signal data. For these reasons, a novel approach is required for the proposed CM.

The solution presented in this thesis is based on the concept of Time Blocks. A Time Block is a fixed period of time over which all signal loss recordings that occur during that period of time affect its average. The actual average signal loss of a cell is the average of the number of Time Blocks being kept (the Averaging Window). Traditional windowing techniques could then be applied on Time Blocks in an Averaging Window, though in the interests of simplicity a rectangular window average is applied. Other windowing implementations are left for future work (see Section 4.6).

Figure 6 presents an example of this. The length in time of each Time Block is 10 seconds, i.e. each Time Block will keep the average signal loss of 10 seconds worth of updates. All updates that will occur are averaged into the latest block (the *current time block*), where the update is added to the current block's sum of signal loss values, and the current block's count value is incremented by 1, to reflect the additional packet that has now been recorded. If an update arrives at a time outside that Time Block (the current block's start time plus a Time Block's length), then a new Time Block is created to represent the current Time Block, and all other Time Blocks are shifted down one place.

In Figure 6, four time blocks are kept at most, which means anything before 1:10 in the diagram is lost. This ensures that data is kept recent, while at the same time ironing out small fluctuations in signal strength that might occur in a short burst (i.e. a few seconds). This aids the CM in countering the effect of varying signal strengths on the Future Neighbours algorithms, which are time expensive and have a chain-reaction effect. The

importance of separating averages into Time Blocks is in sharing those Time Blocks with other nodes. A Time Block which is being updated cannot be shared with other nodes, whereas previous Time Blocks have an average value for signals collected by a single node. These averages for each Time Block for each cell are small enough to share with other nodes on a regular basis, as will be detailed in Section 4.4.2 below. There is a requirement in order to share Time Blocks that all nodes maintain accurate clocks in relation to one another. This is feasible through a number of means, the simplest being provided by the positioning system in use, such as GPS.

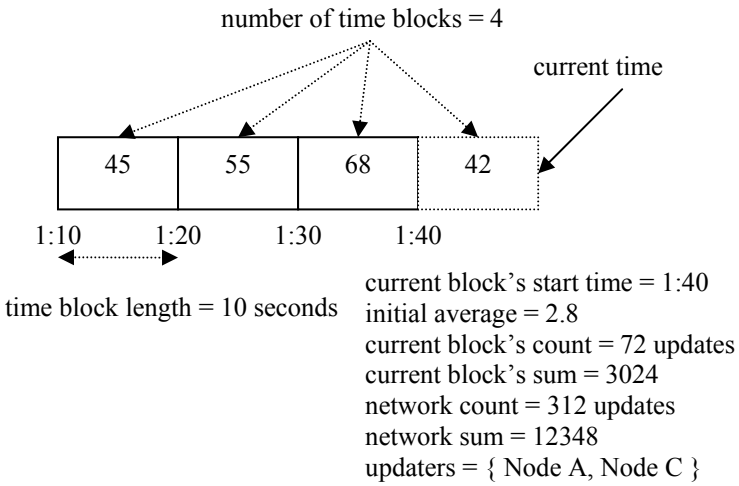

Figure 6: Averaging Window Structure

Each Time Block stores several fields. The *start time* is the period for which this time block takes effect. The *initial average* is the average returned when no values have yet been recorded, but where an average may be required. The *initial average* is always set to the average of all time blocks before the latest time block was created. The current block's *count* and *sum* are used when sharing the map with other nodes, and represent the averages this node itself received and calculated. The network *count* and *sum*, on the other hand, contain all averages, both locally recorded and those obtained from the network as a whole. These are the values used to calculate the average for this cell, and represent the average that the network as a whole produces. The *updaters* field is a set of network addresses of nodes which have already updated this Time Block, so that no node contributes more than once to any Time Block. This is important as maps may be passed often due to map sharing not being an acknowledged process (discussed in Section 4.4.2). Repeated updates may therefore contain some of the same Time Blocks. The *updaters* field solves this potential problem by ensuring that each Time Block from a neighbouring node contributes only once.

4.3.5 Signal loss representation

Signal loss has been used throughout the descriptions above as a means of describing how a signal will propagate over distance. The signal loss (measured in dBm) between two nodes can be easily subtracted from the total power required between those nodes. This can be used to find if the signal is predicted to reach the destination and thus be used by the Future Neighbours algorithms to determine if two nodes at two future locations are predicted to be neighbours.

The total required power takes into consideration both the transmitting output power and the receiving sensitivity values of wireless nodes. Each wireless device has an output power, which can be measured in dBm. Each wireless device also has a receiving sensitivity (also measured in dBm). The CM describes the propagation relation between two identical types of wireless device over different areas of a map. Each CM therefore has some maximum Acceptable Signal Loss value (total required power), *ASL*, that can be calculated as:

$$ASL = p_{out} - s \tag{4.1}$$

where p_{out} represents the output power and s represents the receiving sensitivity [Sha01].

This provides a CM with a maximum power that will decrease over distance and that must be greater than zero in order for it to be received by another listening node. The CM can calculate the estimated signal loss between two locations, and subtracting this from the maximum ASL provides a prediction of whether those two nodes will be able to communicate.

Thus, along with maximum ASL, a calculation for estimating signal loss is required. The fundamental purpose of the CM is to represent signal loss over physical space. Signal loss is extremely hard to predict and calculate. In perfect space with no interference, often called *free space*, an electric burst of a certain power and on a certain frequency will lose power at an exponential rate as it radiates in all directions from the source [Sha01] (refer to Figure 7 on the following page). A signal in reality does not travel only as straight paths radiating out from the source at every angle, but may bounce off objects in the environment, forming many paths to a destination. This thesis looks only at direct signal paths, known as *line of sight*, where a signal can travel perfectly without interference

towards the destination. Multi-path signal propagation is outside the scope of this thesis (see Section 4.6).

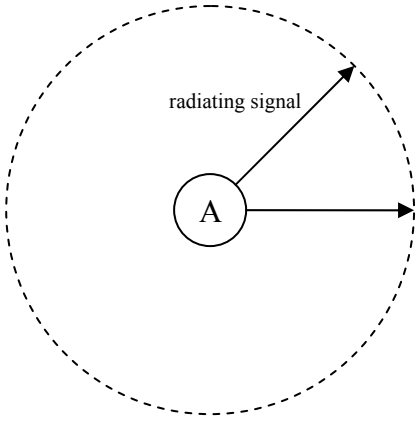

Figure 7: Signal Propagation

The general formula for calculating free space loss, S_{loss} (dBm), in ideal circumstances of wireless signals [Sha01] is:

$$S_{loss} = 32.4 + 20\log_{10}F + 20\log_{10}D \qquad (4.2)$$

where F is the frequency (MHz), and D is the distance (km) between the two nodes.

This formula, however, does not take into consideration different signal propagation models over varying areas. The CM is centred around the concept of cells. A signal may pass over any number of cells, with different values representing the average signal loss over those areas. Figure 8 is presented as an example. In this example a signal between two locations passes through two cells, B and C. An incorrect assumption to finding the signal loss of both areas is to find the signal loss of the distance the signal travelled in cell B (with some modifier adjustment for the signal loss specific to cell B), and then find the signal loss of the distance travelled in cell C, and subtract both of them from the maximum ASL. This is essentially creating 2 radiating patterns (refer to Figure 9), which is an incorrect *estimation* of how the signal would be expected to propagate in reality (Figure 10).

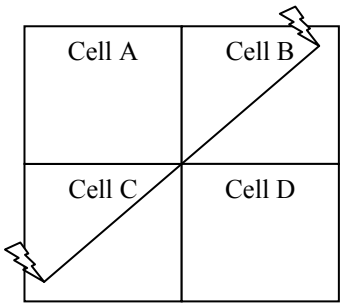

Figure 8: Example Signal

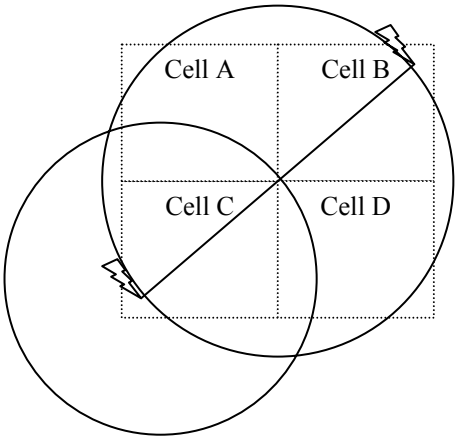

Figure 9: Incorrect Usage of Signal Loss Formula

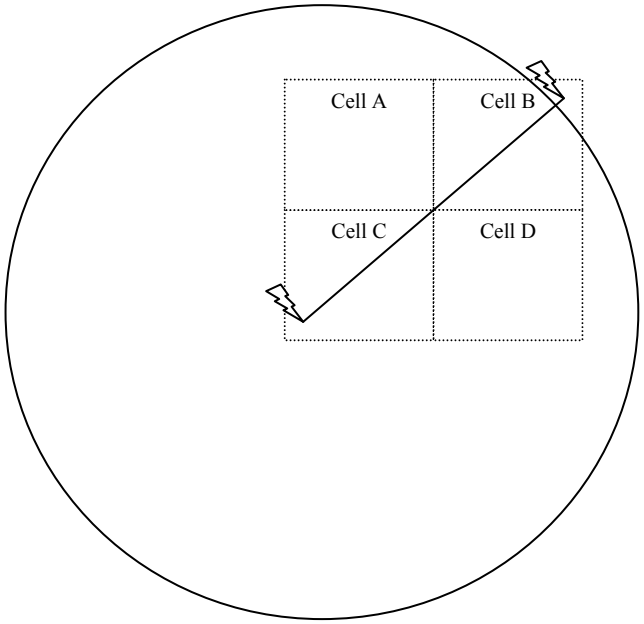

Figure 10: Correct Usage of Signal Loss Formula

The free space loss formula may only be used *once* for the entire signal. To have an effect on the overall loss of a signal, each line *segment's* distance (how much a signal travels within a cell) needs to be adjusted by some modifier which represents a cell's effect on signal loss. To overcome this problem, the concept of *logical distance* is introduced. Any signal received or predicted using the CM is based on a *logical distance*. The *logical distance* that a signal travels is the distance it would need to *physically* travel in order to produce the same amount of loss, thus allowing the free-space loss formula (Equation 4.2) to be used with given multiple signal modifiers.

4.3.6 The *Modifier* Metric

Each cell represents the average signal loss of signals passing through that cell. A cell must be able to apply the average signal loss to the distance a signal is predicted to travel through that cell. In other words, the cell needs some way of applying its signal loss average representation to a *physical distance* in order to produce a *logical distance*. This thesis presents a straightforward approach to this problem, the *modifier* value. The value stored for each cell is the *modifier* that a signal applies to the *physical distance* of a signal as it passes through that cell, which when multiplied with the *physical distance* creates a *logical distance*. The minimum modifier value is 1.0, in other words a *logical distance* is identical to the *physical distance*, and thus represents perfect free-space loss. The *modifier* of each cell is used to extend the distance of a signal to the distance it would need to travel in perfect free space to achieve the same loss.

To calculate a predicted signal loss between 2 known points on a map, a sum of the *estimated logical distances* of all cells between the 2 points is calculated and applied to the formula above. The *estimated logical distance*, *ELD*, from any cell is calculated as:

$$ELD = d \times m \tag{4.3}$$

where d represents the distance that the signal passes through the cell, and m represents the cell's signal loss *modifier* value.

Where a cell on the CM has subdivisions, it is the subdivisions that are evaluated, and not the parent cell. Only *actual cells* are used for estimating signal loss, as they are the lowest level of accuracy that the CM has instantiated.

For an example of this process, consider Figure 11 on the following page. A signal is predicted to travel from Node A to Node B over the given CM. The direct line between the two nodes is formed, and a list of cells over which the signal will pass is created. Each of these cells multiply their *modifier* by the *physical distance* that the signal travels through their cell. The resulting *logical distance* is used to find the signal loss, which can consequentially determine whether two nodes are predicted to be neighbours.

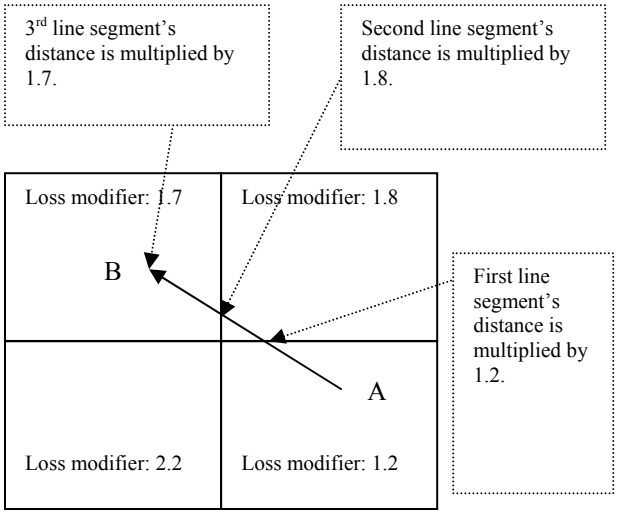

Figure 11: Signal passing through multiple cells

This formula uses the *modifier* of each cell to *extend* the distance of a signal to the distance it would need to travel in perfect free space to achieve the same loss. While this is not completely accurate, a perfect model is not required to achieve reasonable predictions (refer to Chapter 8 for experiments).

4.3.7 Estimating Signal Loss

Predicting signal loss is relatively easy to perform. Given any two physical locations, the CM forms a line between those, and any cells that the line passes through add to the total *estimated logical distance* (*ELD*) of the signal. A cell's average *modifier* is multiplied by the distance the signal travels through the cell to calculate its addition to the *ELD*. Applying the free-space loss formula (Equation 4.2) to the *ELD* gives the signal loss in dBm, immediately applicable to maximum ASL levels to determine communication capability. Algorithm 4.2 details this procedure below.

Algorithm 4.2: Estimate Signal Loss

- Let *sl* represent the line of travel which a signal is predicted to travel.
- Let *s* represent the list of cells in a CM over which *sl* intersects.
- Let *eld* represent the Estimated Signal Loss which this algorithm will calculate.

1. Iterate over each cell, *c*, in *s*:
   - Let *m* represent the cell modifier of *c*.
   a. If the cell, *c*, at this position does not yet exist, create it with a default modifier of an average of the cells surrounding it.
   b. Calculate the distance, *d*, of which *sl* is within the bounds of *c*.
   c. Add to *eld* the logical distance from *c*, using Equation 4.3 (given *m* and *d*).

Cells are created for any physical space over which a signal loss estimation is required but as of yet is not mapped. CMs will expand as either received signals are mapped or as signal losses are estimated. One of the challenges presented in Section 4.2 is in estimating signal loss for unknown and unmapped areas. The CM will initialise a cell's default modifier to an average of the existing surrounding cells (each cell is surrounded by eight other cells, though they may not necessarily exist yet). The default modifier is only used until the first signal is mapped over a cell. It exists to provide estimations of signal loss over unknown areas based on the signal propagation characteristics of surrounding nodes.

4.4 Map Creation

This section describes the process of generating a signal loss map based on the CM. To estimate the loss of a signal through various cells, the modifiers of each cell are built up over time as actual signals are received and mapped over the appropriate cells. It would be rare that all cells will be covered by a single node, and certainly never all at the same time. The CM is therefore shared between nodes in order to build a more comprehensive analysis of the surrounding logical environment. Thus there are two ways to update a CM: using actual received signals and merging CMs from other nodes. Figure 12 shows the two methods by which a node's CM may be constructed, and how this is then used to provide signal loss estimations to other algorithms (such as the FNT algorithms).

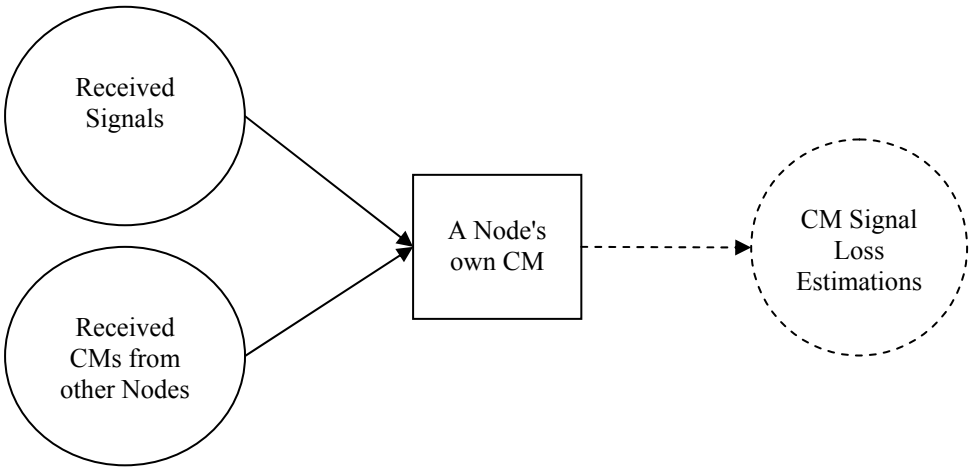

Figure 12: CM Usage

4.4.1 Using Signals

From a network-centric point of view, the entire CM is developed from the signal loss readings available with each incoming packet. While each individual node develops its own copy of the map using a combination of signals and map updates, it is the construction of the map from received signals that ultimately creates signal loss maps.

Every signal received by a node is used to update the CM to increase its accuracy. Each time a packet arrives from another node, it carries with it the coordinates of the node from whence it came. This is then combined with the coordinates of the current node, forming a path over which a signal is predicted to have travelled. It is impossible to calculate the logical distances for each cell from the given information with precise accuracy. What this research proposes is that a signal will be mapped across the *probable* cells which are likely to have affected the signal, by assuming the signal travels in a straight line.

For each incoming packet, a Signal Status Event is generated and passed on to the CM's update routines. Each Signal Status Event contains two important fields, a *path* and a *signal loss*. The *path* is a single line between the location of the receiving node (the current node building this map) and the sending node. The *signal loss* is the measured amount by which the original signal was degraded. The signal loss is derived from knowledge of the sender's transmitting signal strength and the received signal strength. The received signal strength can be obtained from 802.11b wireless interfaces for each incoming packet, while the sender's transmitting strength must be manually input based on the hardware used for nodes. While this is not optimal, there are presently no other solutions for determining the sender's transmitting signal strength. Using these two fields, the received *signal loss* is mapped over the cells that the signal's *path* crosses.

One of the difficulties in mapping signals given only a signal's assumed path of travel is that a signal line does not directly indicate where loss is more prevalent. Cells are likely to have different contributing factors to signal loss, and thus future signal loss should be mapped accordingly. For example, in Figure 13 on the next page, a signal will be mapped over two cells, one with a very high modifier of 2.0, and another cell with a very low modifier of 1.0. If a weak signal is mapped across these cells, the modifiers of both cells will increase. Even though the signal's loss is high, most of the loss should be attributed

to the cell with the high modifier. In other words, both cells will not receive an equal amount of the signal's loss simply because the distance travelled through both cells is roughly equal. Because one cell has already been calculated as having a high signal loss, it should have a higher portion of the loss attributed to it when further signals are mapped. In this way it is also possible to derive the approximate signal loss modifier of cells without nodes venturing into them. Any cells between two cells that two nodes reside in can have their modifier derived based on the modifiers of known cells.

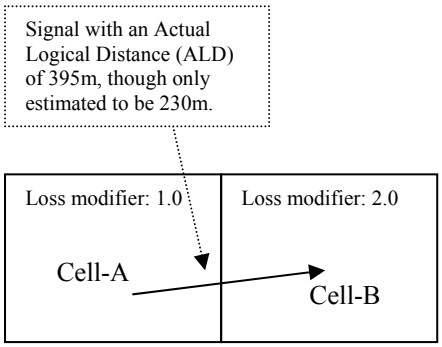

Figure 13: Example usage of Weight

A simple approach is presented in this thesis to more accurately map signal loss. A portion of the signal's loss is allocated to each cell equivalent to how much the cell in question is likely to have influenced the given signal (using the *modifier*). The principle of a cell's *weight* is introduced. A cell's *weight* on an incoming signal is the percentage of signal loss that a cell contributes to an *estimated* signal between the same two points as an actual signal (Equation 4.5 below). The percentage of loss from an estimated signal is more likely to be accurate than assuming all cells contribute equally based on distance travelled through each cell alone. Though in reality it is impossible to ascertain for certain where signal loss occurs from a simple signal line, this approach provides a means of improving signal loss attribution given the information available to nodes (refer to Section 4.6 for issues outside the scope of this thesis). As information is received across multiple cell combinations, cell accuracy should improve. As the actual cells closer represent their real modifiers, future signals are mapped more accurately to the appropriate cells, creating a more accurate CM overall.

To map an incoming signal and its loss over cells, Algorithm 4.3 is presented. The following terms are used in the algorithm:

- (signal | cell) Physical distance (PD) :
  - [signal]: The physical distance between two nodes.
  - [cell]: the physical distance between the two points of a cell that a signal passes through.
- Logical distance (LD) : The physical distance a signal would need to travel in free space to produce the same loss as an actual signal passing through a cell or cells.
- Estimated logical distance (ELD) : The predicted logical distance that a signal or cell produces given a physical line via which a signal is assumed to travel. This is the same value that would be produced when estimations of the logical distance between 2 points are required by any estimation routines.
- Actual logical distance (ALD) : The logical distance value that represents the loss that a signal actually produced. This value is calculated through the reverse free-space loss formula (Equation 4.4), when a signal is received and will be processed.
- Modifier : A cell's current signal loss multiplier that it applies to the physical distance that a signal travels through a cell in order to produce a logical distance.
- Weight : A fractional value of how much a cell presumably has influenced a received signal. This is calculated by dividing the *cell's* estimated logical distance by the *signal's* estimated logical distance.

The following equations are defined. The equation to calculate the logical distance, *LD* (in kilometres), of a given signal loss is:

$$LD = 10^{\left(\frac{S_{loss} - 32.4 - 20 * Log_{10} F}{20}\right)} \tag{4.4}$$

where F is frequency (in MHz), and S_{loss} is the recorded signal loss of the original signal between the two nodes (in dBm). This is a rearrangement of Equation 4.2 earlier.

To calculate the weight, W, that a cell has on a given signal:

$$W = \frac{C_{eld}}{S_{eld}} \tag{4.5}$$

where C_{eld} is the cell's ELD, and S_{eld} is the signal's ELD.

Define N_{ld}, the new *LD*, to be:

$$N_{ld} = S_{ald} W \tag{4.6}$$

Define N_m, the new *modifier*, to be:

$$N_m = \max\{\frac{N_{ld}}{C_{pd}}, 1.0\} \tag{4.7}$$

where S_{ald} is the signal's *ALD*, W is the cell's *weight*, and C_{pd} is the cell's *PD* travelled.

Following is the algorithm to map a received signal in a Signal Strength Event over the affected cells.

Algorithm 4.3: Map Signal Loss

1. Obtain a list of cells over which a signal was assumed to travel, and the *PD* through the cell that the signal passed through.
2. Calculate the signal's *ELD* using Equation 4.3
3. Calculate the signal's *ALD* using Equation 4.4
4. For each cell, *c*, calculate:
   a. The *ELD* using Equation 4.3
   b. The *weight* using Equation 4.5.
   c. The new *LD* using Equation 4.6
   d. The new *modifier* value using Equation 4.7.
5. Order the list of cells in ascending order based on the new *modifier* value.
6. Recalculate equations 4.5 to 4.7 for each cell using the signal's *ALD* and *ELD* values that are adjusted after each cell iteration. The adjustments after each iteration are:
   a. The cell's ELD is subtracted from the signal's ELD.
   b. The cell's ALD is subtracted from the signal's ALD.

Algorithm 4.3 is executed twice per incoming signal. First on all actual cells that a signal passes through and then on all potential cells that a signal passes through. Actual cells are always used to find signal loss estimates through the CM, while potential cells monitor the need for an actual cell to subdivide. The layer of *actual cells* is the level of detail the CM uses to adequately and efficiently represent the surrounding logical communication topology. The algorithm is repeated on the level of detail below this layer, the *potential cells*, so that the CM can later calculate whether there is enough of a difference between a cell's subdivided *potential cells* to warrant a subdivision.

The overall *LD* is divided among each of the cells that are assumed to have influenced the signal received for the Signal Status Event. The *weight* in step (4b) represents how much a cell affects the signal overall. This helps ensure that the incoming signal applies any negative loss to cells that have an average which indicates that they are already poor signal strength cells. If the new *modifier* calculated in step (4c) is less than 1.0, then the cell's current modifier has dropped below realistic levels. Though not impossible [AB25], it is highly unlikely that a signal will ever be naturally amplified in a daily environment, and consequently a signal should not travel with less loss than free-space loss. The new modifier is kept at 1.0 if this situation occurs, and the cell's new logical distance is set to the cell's physical distance travelled. This artificial raising of modifiers to a value of 1.0 simply enforces a minimum modifier value.

Steps (5) and (6) are consequently made to adjust any modifiers that were calculated below 1.0 and have been instead maintained at 1.0. Because of this change, other cells need to carry the burden of this effect, by lowering their own new modifiers accordingly. For those cells that have boosted new modifier values, in step (6) they will subtract larger *ALD* values than their previous *weight* values had calculated. With a diminished signal's *ALD*, further iterations of this process on other cells will affect them accordingly, so that the signal is balanced as accurately as possible over the affected cells.

As an example of the entire process, refer to Figure 14 and Table 4.1 below. An incoming signal is mapped across two cells, Cell-A and Cell-B. This signal has an ELD of 230m yet an ALD of 395m, indicating that the incoming signal has a greater loss than the two cells' current average. Cell-B's modifier and the distance the signal travels through it are both higher than Cell-A, consequently Cell-B has a higher calculated *weight* on the signal, 0.7 (70%). Both cells have an increased LD to account for the loss, leading to increased modifiers for both cells. Because of Cell-B's high *weight* on the signal, Cell-B's resulting modifier is higher than Cell-A's. These new modifiers will be averaged into the most recent Time Block of each of the two cells respectively. With repeated signal readings, both cells will alter their signal loss modifiers until they accurately represent the signal propagation characteristics through their respective cells as best as possible.

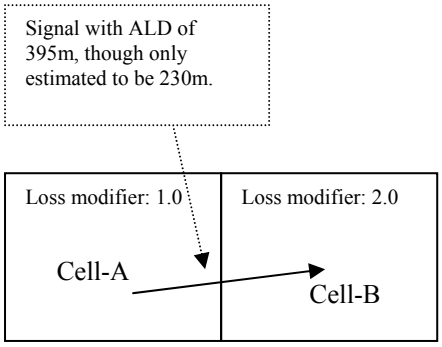

Figure 14: Example Use of Algorithm 4.3

Table 4.1: Sample Calculations for Algorithm 4.3

1. Find the list of cells		**Cell-A**	**Cell-B**	**Total**
physical distance of signal		70m	80m	150m
current modifier		1.0	2.0	
2. Calculate signal's ELD		70m	160m	230m
3. Calculate signal's ALD				395m
4. Calculate each cell's *weight, new logical distance* and *new modifier*	***b. weight* on the signal**	0.30 (70 / 230)	0.70 (160 / 230)	
	c. new *LD*	395m * 0.30 = **118.5m**	395m * 0.70 = **276.5m**	
	d. new *modifier*	118.5m / 70m = **1.69**	276.5m / 80m = **3.46**	
5. Order the list of cells by their *new modifier* in ascending order		1st	2nd	
6. Recalculate each cell's *weight, new logical distance,* and *new modifier*		(no changes compared with Step (4))		

The emphasis of this algorithm is to provide a reasonably accurate mapping of a signal over the cells that most likely influenced it. Chapter 8 will present experiments validating Algorithm 4.3's signal loss mapping technique compared with a generic distance-weighted technique. Given that any combination of cells may be part of a signal mapping calculation, only a large quantity of signals covering multiple cells in problematic areas on a map will be able to increase the accuracy of individual cells. Calculating a cell's signal propagation characteristics when nodes may never even enter a particular cell is too complicated for a signal loss map optimized for automated wireless ad hoc devices.

4.4.2 Merging CMs

Communication Maps are more accurate when data is shared with other nodes. Previous signal strength measurement experiments [HSS03] have concluded that raw signal strength measurements are consistent between mobile nodes that have identical hardware, simplifying this process. As such, each node is able to broadcast its CM as a map of cells, each cell containing a list of previously completed Time Blocks. This is broadcast as a Map Update Packet (MUP).

The basic premise of sharing CMs among nodes is that only each node's own received signal calculations will be shared with other nodes. Consider three nodes, A, B, and C, which are all ready to broadcast their CMs. Node A broadcasts its CM first, which is received by nodes B and C. If the CM received by Node B is used to update Node B's own CM before it broadcasts its own to nodes A and C, then both A and C will receive a collation of not only Node B's CM, but Node A's as well. This is the reasoning behind keeping individual and network averages for each Time Block separate, as described earlier in Section 4.3.4, to avoid each node receiving repeated averages.

In sharing maps, it is important to weight each node's contribution relative to its perceived value. While every node may have an average for a particular cell in their CM, nodes may not have the same quantity of samples recorded as other nodes. The approach used in this thesis is to allow each received signal loss value to be weighted as equally as any other received signal from any shared CM. This is accomplished by adding further details to the Time Block implementation. Each Time Block average is kept in the detailed form of total modifier sum and the count of modifiers received. This addresses the concern of weighting each received CM accurately. Instead of each transmitted CM sending an overall average for a Time Block, the CM sends the total of all signal loss samples within that time frame, along with the number of samples recorded. Combined, the two present an accurate view of how much data a given node has collected, and thus the weight of importance that a receiving node of a CM should apply to the received data. This approach is not without its flaws. Some samples may theoretically have a greater impact on accuracy than other samples, depending on node movement and position from each cell. However, such details are beyond the scope of this thesis, and considered too computationally and memory intensive to be part of the CM algorithm (see Section 4.6).

The final requirement of map sharing is to ensure that each completed Time Block of each cell be counted only once per receiving node. This is regardless of how many times a MUP may be broadcast. A MUP may be received multiple times as it is broadcast across a network, due to the nature and range of wireless networks. Each new MUP may contain a small history of recent Time Blocks. This is to overcome lost MUPs or nodes receiving MUPs which have been outside the communication range of the network previously. Both of these problems are addressed at the receiver end of a MUP. Each Time Block of each cell will list the nodes that have currently had successful MUPs incorporated into a Time Block, ensuring each node's contribution is counted only once. This further establishes the reasons behind having multiple Time Blocks, as an active Time Block will never be broadcast. Only once a Time Block has been superseded will it be shared with neighbouring nodes.

The design of each MUP is relatively simple. Each CM has a *size*, the range of cells that it covers. This information is shared with other nodes so that all nodes share a common scope. Referring to the packet format presented in Figure 15, the values *x1*, *y1*, *x2*, and *y2* describe the area that the map covers in some common measurement. This is implementation-dependent, although metre values based on GPS coordinates are used in this thesis. Precision accuracy of GPS readings is another important issue for real world implementations of the CM, but will not be investigated in this thesis. The number of cells field indicates how many cell data structures will follow, followed by the cells themselves.

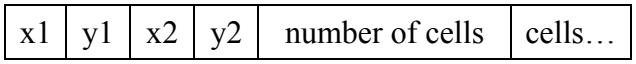

Figure 15: MUP Format

Each shared cell has its own data structure, describing the nature of the cell and within it any subdivided cells that form part of the cell. This data structure is shown in Figure 16. Every cell has an x and y position (latitude and longitude GPS coordinates), and a pair of size values (all described in metres). The cell type field indicates what type of cell this is, either a parent cell, actual cell, or potential cell. The averaging time window (repeated in Figure 17 from earlier) is a list of averages collected over time. The number of time blocks and the length of each time block are constant for the entire network, making the

size of the averaging time window fixed. For example, if the CM is configured to collate four Time Blocks, then three Time Blocks will be broadcast each time (all but the latest active Time Block). For each Time Block, the block's start time, a sum of signal loss multipliers recorded, and a count of signal loss multipliers recorded are stored. If the cell type field is a parent cell or an actual cell, then there will be four subdivided cells of the same structure following the averaging time window. This process repeats for the depth of subdivisions.

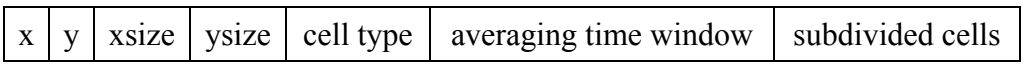

Figure 16: MUP Cell Format

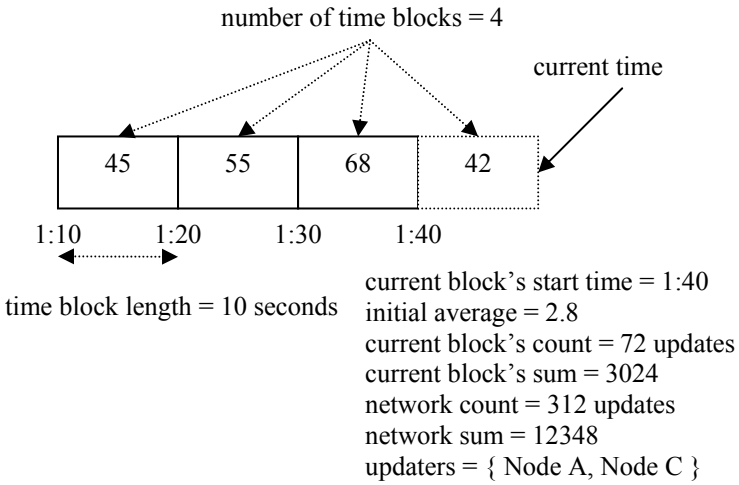

Figure 17: Averaging Time Window Structure

When a MUP is received by neighbouring nodes, each of these nodes performs a common algorithm, listed in Algorithm 4.4. All nodes will share a common map size so that the physical area covered by all CMs is the same, ensured by step (1) of each MUP. All nodes share initial *Default Cell Size* (*DCS*) cells with a common alignment, so that the top layer of a map can always be shared. For example, if the DCS is 100m, then geographical coordinates (e.g. from GPS) are aligned to 100m boundaries. Only if a level of detail exists in a receiving node's CM will it use any level of detail (through subdivided cells) of a received MUP (steps (2a) and (2b)). As *potential cells* are shared, and all *actual cells* have *potential cells*, all nodes will eventually have the same overall average modifier if all MUPs are received successfully.

A cell will only be updated from a MUP if it exists in the local CM. At the top-most level, each CM will be expanded to the size of any received MUPs, so that the area covered by

all CMs is equal. However, if the MUP contains subdivided cells that the local map has not subdivided, then subdivisions will not take place. To do so would make redundant the cell subdivide and merge algorithms (Algorithm 4.1), as subdivisions could exist that are not justified by the local CM. If a local CM receives MUPs from all other nodes in the network, then subdivisions will eventually take place if they are required, as the signal loss averages for a cell in all CMs will be the same. Timing of received MUPs could change the order in which subdivisions occur, but the subdivisions themselves will eventually take place. If MUPs are not received, then CMs may vary. However, given that cells are averaged over time, if signal loss is evident in an area then repeated MUP transmissions will eventually propagate this information to all nodes.

Algorithm 4.4: Process a Received MUP

1. Expand the size of the CM if required to cover the area indicated by the MUP.
2. Iterate over each cell in the MUP. For each cell:
   a. Iterate over each T*ime Block* in the *averaging time window* of the cell from the MUP. If an equivalent T*ime Block* exists in the receiving node's cell, and the node transmitting the MUP has not already updated this T*ime Block*, then the Time Block's *sum* and *count* are added to the receiving node's *network sum* and *network count* fields, and the MUP's transmitting node address is added to the *updaters* list.
   b. Repeat the process for each subdivided cell, if it exists in both the local and the MUP's cell.

This section has detailed the two processes in creating Communication Maps for a node, using a node's own signal analysis and that of the network as a whole through map sharing. This produces as detailed a map as possible using only ad hoc wireless nodes. Communication overhead of map sharing as well as signal estimation accuracy will both be tested in Chapter 8. The next section identifies an alternate approach to creating Communication Maps with the addition of *boundaries*.

4.5 Boundaries

The purpose of the Communication Map is to represent various areas as having similar signal propagation characteristics. However, many real life situations involve closer levels of detail which are too fine for the CM to represent, such as a building wall. A common building wall has a very high signal loss, which is why "line-of-sight" signal propagations transmit far further than intra-building signals. These objects are too thin to be mapped by a CM, and their effect is instead incorrectly represented by an entire cell.

Consider the example given in Figure 18. A simulated wireless map is created with 4 stationary nodes placed across two cells, with nodes C and D on the left and in an open-air, free-space loss environment. Nodes A and B are placed inside a simulated large empty building (like a shed), with walls having 150 meters of logical distance loss (83.5 dBm loss), but within the building there is free-space loss.

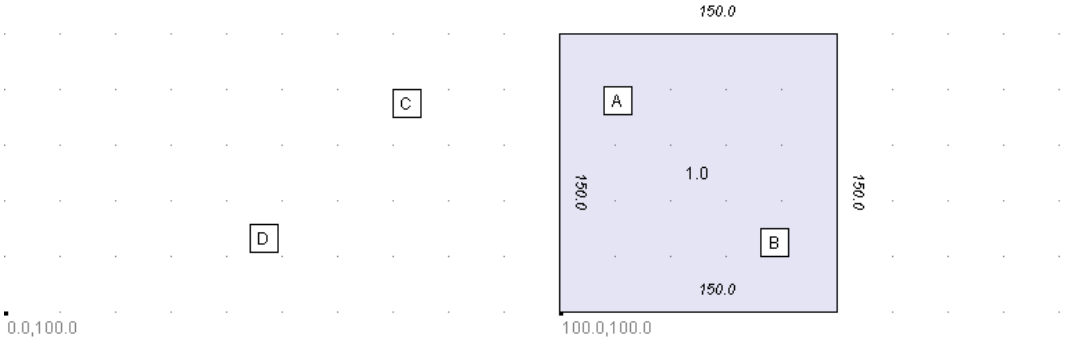

Figure 18 : Simulated Wireless Map

The CM categorises areas as having a particular signal loss, starting with reasonably large cells and subdividing them as more detail is required. The results from the above experiment lead to nodes delineating on both cells a poor signal loss multiplier. Figure 19 shows an example of this with Node D's CM after 25 minutes using an implementation of the CM algorithms presented earlier. As signal loss is not localised to any smaller area, no subdivisions take place.

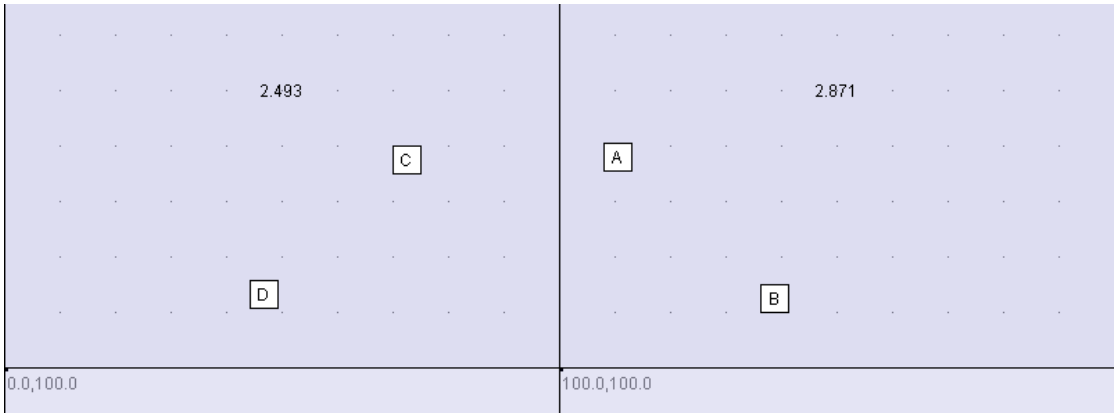

Figure 19 : Node D's Communication Map at 1500 seconds

While this *averaging* of areas as a whole performs as expected, it is interesting to see the effects of attempting to integrate finer objects into the CM. An alternative approach is to incorporate the concept of boundaries to each cell. In order to support real-life situations

such as buildings, the CM must be able to identify that communication within a cell may have a different signal loss to signals leaving that cell in a particular direction.

4.5.1 Design

In this research there is a need to represent signal loss which occurs due to thin objects. These objects (such as walls) are generally not thought to be able to contain nodes, rather nodes will exist outside these objects, in regular cells. The solution presented is to add boundaries surrounding each cell that represent immediate signal loss for each signal passing through. Every cell and subdivided cell has surrounding it boundaries that can have a different signal loss modifier than each other and different from the cell's own. With the square-shaped cells used in this work, each cell has 4 boundaries. Figure 20 shows an example of a CM implementing boundaries. Each cell has a signal loss modifier for the area within it, as well as signal loss modifiers for each of the four boundaries of a cell. Adjacent cells always share a boundary, as a signal between cells should not have a different loss.

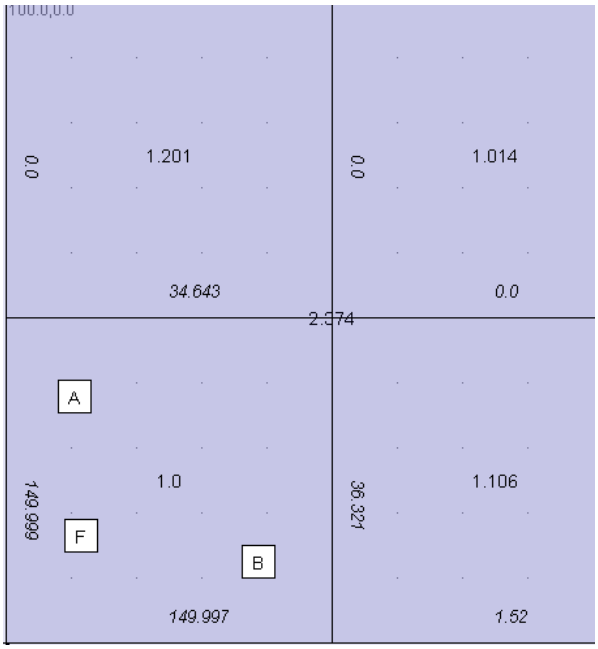

Figure 20 : Cell Boundary Example

The CM is made up of both cells and boundaries. Every side of the cell contains a shared boundary with its adjacent cell in that direction. When a cell subdivides, it creates the extra four boundaries required for the four subdivided cells it creates. To illustrate this, refer to Figure 21 on the next page. Each time a cell is subdivided, it creates four inner

boundaries, and four subdivided cells. Each of the subdivided cells will share two of the inner boundaries with other subdivided cells, and will share two of the original cell's boundaries. These boundaries are not necessarily subdivided. The northern boundary in Figure 21, for example, is shared not only with the cell or cells above it, but also with the two top-most subdivided cells in this cell. This boundary will only subdivide if one half differs from the other more than the *Minimum Boundary Subdivide Difference* (*MBSD*). If the internal cells or inner boundaries cause the cell to subdivide, the outer boundaries may not necessarily have potential subdivisions with a difference exceeding the MBSD, and thus have no need to subdivide also.

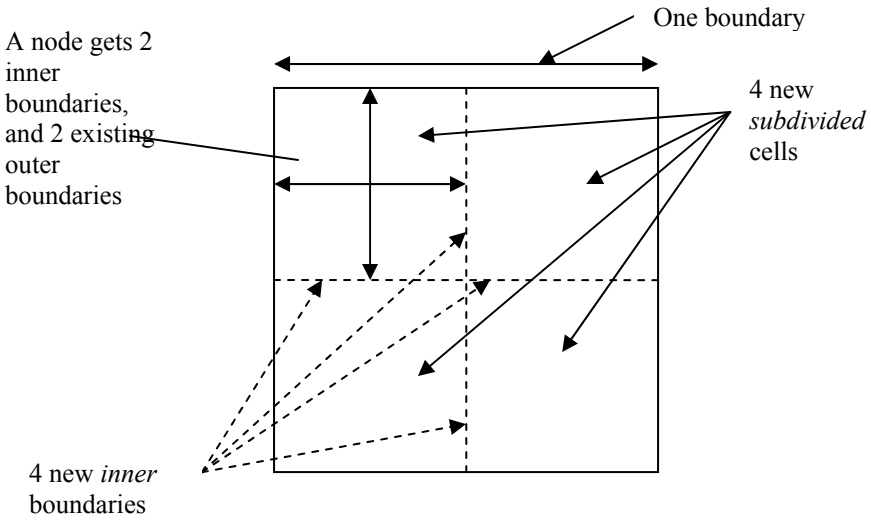

Figure 21 : Subdividing Cells implementing Boundaries

4.5.2 Algorithm Overview

With the introduction of boundaries a number of methods exist for creating and using the CM. Figure 22 presents the algorithm flow for all algorithms associated with the CM. There are three algorithms which provide the CM with signal loss measurements, Algorithm 4.3 (using signals based on cells), Algorithm 4.7 (using signals based on cells and boundaries) and Algorithm 4.4 (using a shared map from another node). After the CM is updated from an input method, subdivisions are evaluated using either Algorithm 4.1 (to evaluate cell subdivisions) or Algorithm 4.5 followed by Algorithm 4.6 (to evaluate boundary and cell subdivisions respectively). Finally, Algorithm 4.2 can be used to estimate signal loss based on the CM.

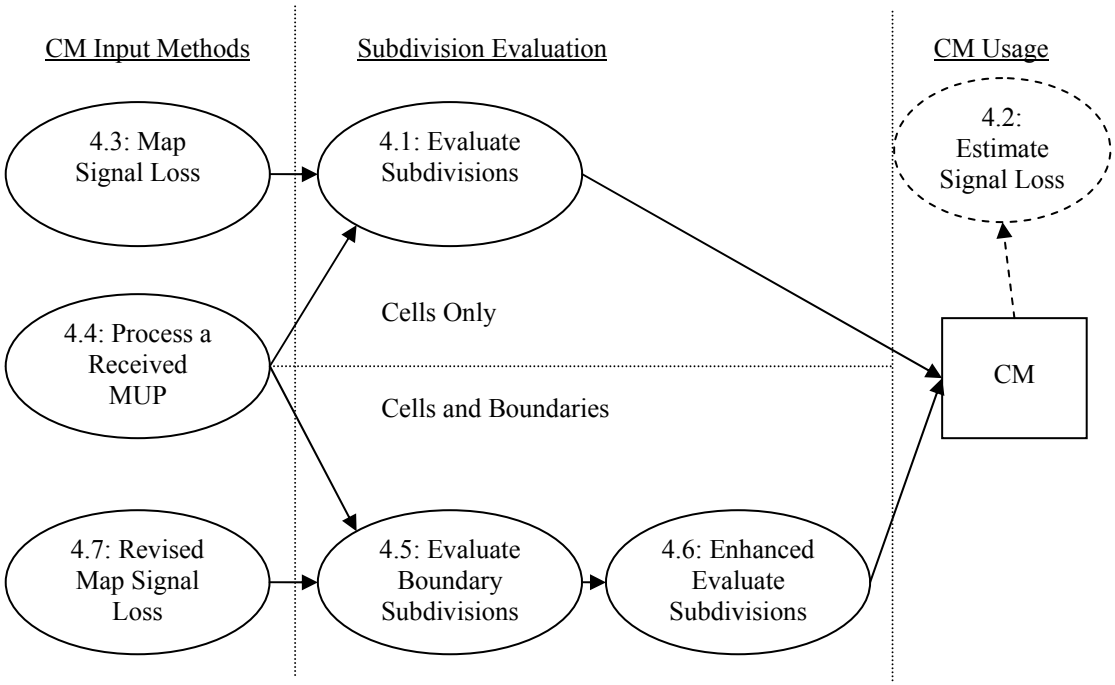

Figure 22: CM Algorithm Flow

4.5.3 Evaluating Subdivisions

Two algorithms are used to determine the need to subdivide or merge communication objects. The first algorithm, Evaluate Subdivided Boundaries, is described in Algorithm 4.5. It is similar to Algorithm 4.1 in that each pair of subdivisions is analysed for the need to subdivide or merge. The algorithm first recursively traverses to the lowest level of subdivision before determining whether each pair of boundaries is eligible to subdivide or to merge, based on the MBSD and Minimum Boundary Merge Difference (MBMD) values. Boundaries will not subdivide further than the MCS. The MCS value is the minimum length an object can be (in metres), and applies to boundaries in the same way as it applies to cells.

Algorithm 4.5: Evaluate Boundary Subdivisions

- Let *b1* represent the one of the subdivided boundaries.
- Let *b2* represent the other of the subdivided boundaries.

1. Iterate over both subdivided boundaries:
   a. If the subdivision is not a potential boundary, repeat Algorithm 4.5 for the subdivided boundary.
2. If the boundary type is an actual boundary and the difference between *b1* and *b2* is greater than the *MBSD*, and the *MCS* hasn't been reached, convert the boundary into a parent boundary.
3. Otherwise, if the boundary type is a parent boundary and all subdivisions are actual boundaries *(no parent boundaries)* and if the difference between *b1* and *b2* is less than the *MBMD*, convert the boundary to an actual boundary.

The original algorithm to evaluate subdivided cells (Algorithm 4.1) needs to be modified as each cell now has boundaries. The enhanced algorithm to evaluate subdivided cells is described in Algorithm 4.6. This algorithm is modified such that the highest and lowest signal losses of the inner boundaries are found. A cell will subdivide if the difference between the highest and lowest signal loss from cells or from boundaries is greater than the MCSD or MBSD values respectively. Similarly, four subdivided cells can only merge into a single cell if both the difference between subdivided cells and the difference between subdivided boundaries are both less than the MCMD and MBMD values respectively.

Algorithm 4.6: Enhanced Evaluate Subdivisions

- Let *hc* represent the highest signal loss of subdivided cells.
- Let *lc* represent the lowest signal loss of subdivided cells.
- Let *hb* represent the highest signal loss of inner boundaries.
- Let *lb* represent the lowest signal loss of inner boundaries.

1. Iterate over each subdivided cell, finding the highest signal loss (*hc*) and lowest signal loss (*lc*) of all subdivided cells:
   a. If the subdivision is not a potential cell, repeat the evaluate subdivided cells algorithm for the subdivided cell.
   b. If the subdivided cell's average signal loss, S_{loss}, is higher than *hc* , let $hc = S_{loss}$.
   c. If S_{loss} is lower than *lc*, let $lc = S_{loss}$.
2. Iterate over each inner boundary, finding the highest signal loss (*hb*), and lowest signal loss (*lb*) of all inner boundaries:
   a. If the inner boundary's average signal loss, B_{loss}, is higher than *hb* , let $hb = B_{loss}$.
   b. If B_{loss} is lower than *lb*, let $lb = B_{loss}$.
3. If the cell type is an actual cell and the difference between *hc* and *lc* is greater than *MCSD,* OR the difference between *hb* and *lb* is greater than the *MBSD*, and the *MCS* hasn't been reached, convert the cell into a parent cell.
4. Otherwise, if the cell type is a parent cell and all subdivisions are actual cells and if the difference between *hc* and *lc* is less than *MCMD*, AND if the difference between *hb* and *lb* is less than *MBMD*, convert the cell to an actual cell.

4.5.4 Mapping Signals

In order to build up a CM's cells and boundaries, the algorithm to map received signals across the map is altered. This algorithm is updated significantly in order to calculate and represent the addition of boundaries. In addition to the terms previously listed for Algorithm 4.3, the following new term is defined:

- Communication Object : A cell or a boundary through which a signal passes. These may be *actual* or *potential* objects, depending on which iteration of the algorithm is being executed.

Algorithm 4.7 maps a received signal in a Signal Strength Event over the affected cells and boundaries.

Algorithm 4.7: Revised Map Signal Loss

1. Obtain a list of cells over which a signal was assumed to travel, and the *PD* through the cell that the signal passed through.
2. Calculate the signal's *ELD* using Equation 4.3.
3. Calculate the signal's *ALD* using Equation 4.4.
4. Create the *Update Modifiers Table* for each cell and boundary in the list of *communication objects*.
5. Sort the list of cells by their new *modifier* value, in ascending order.
6. Let B_{count} be the number of boundaries with an *ELD* of 0.
7. Iterate over the sorted *Update Modifiers Table*:
   a. If the communication object is a boundary and has an *ELD* of 0, and the signal's *ALD* is greater than its *ELD*, give the boundary's *modifier* an *initial bonus* using Equation 4.8.
   b. Otherwise recalculate the *modifier* similar to Step 4 (using Equation 4.7 for cells, and using the ALD for boundaries, as explained below).
   c. Subtract the *ELD* and *ALD* calculated for the entry in the *Update Modifiers Table* from the remaining signals' *ELD* and *ALD* values respectively.

The above algorithm in executed twice, firstly on all potential communication objects that a signal passes through, and then secondly on all active communication objects. The *potential* objects determine the need for an actual cell or boundary to subdivide. The *actual communication objects* also need to be updated, as signal estimations are based on *actual* objects only.

The Update Modifiers Table is a detailed table of information about each communication object (cells and boundaries) that a signal travels through. The table has a row for each communication object, be it a cell or a boundary. Four fields are calculated for each row:

1. Estimated Logical Distance (ELD).
2. Weight.
3. New Logical Distance (new LD).
4. New modifier.

The *ELD* is the logical distance that an object estimates it has added to a given signal passing through it. For cells, this is calculated as the physical distance through the cell multiplied by the cell's modifier. For boundaries, it is simply the modifier value itself, as a signal either passes through a boundary or it does not.

The *weight* field is the weight that a communication object has on the overall estimated signal loss, and consequentially the presumed weight it would have on an incoming signal. It is simply the communication object's ELD divided by the signal's ELD. The *weight* is how much a cell affects the signal overall. This is an important step in ensuring that the incoming signal applies any negative loss to cells that have an average which indicates that they are already poor signal strength cells.

The *New Logical Distance* (new LD) is the logical distance that a new signal is assumed to have had when passing through the communication object. This assumption is based on the signal's *ALD* and the communication object's weight given the *ELD*. For cells and boundaries, the *new LD* is calculated as the object's *weight* multiplied by the signal's *ALD*. There are two exceptions to this. When creating the table (step (4) of Algorithm 4.6), if the new *modifier* calculated for a cell is less than 1.0, then the cell's new *modifier* value is set to 1.0, and the cell's *new LD* is set to the cell's *physical distance* travelled. This is the minimum value that will be enforced in the implementation of the CM.

The second exception is in the recalculation stage of the table (step (7) of the algorithm). If a boundary has a modifier of 0, and the signal's ALD is greater than its ELD, then the boundary is assigned what will be known as an *initial bonus*. Since boundaries do not necessarily exist, all boundaries' modifiers begin at 0. All cells that a signal passes through, even in free-space loss (where the cells have a modifier of 1.0), will have an effect on the signal, even if it is just the distance travelled through a cell. Consequently all cells along the path of a signal will have a *weight* value. Boundaries, however, only exist if there is some object that affects signal loss but is too thin to be mapped in a cell. Therefore all boundaries' modifiers start at 0, which means that initially no boundary has any *weight* on an estimated signal. If a boundary has no *weight*, then under normal operation of the algorithm a boundary will never be given any portion of signal loss. The solution to this

problem implemented in this algorithm is to give a *initial bonus* to a boundary under two conditions:

1. The boundary currently has no effect on the estimated signal (has a *modifier* of 0).
2. The signal's ALD is greater than it's ELD.

Given these two conditions, a boundary will only be given a bonus if it currently does not effect a signal loss estimate, and if the signal's loss was greater than predicted. This means that there is more loss than anticipated, and it should be accountable somewhere (quite possibly by a boundary). After several experiments with this concept (detailed in Chapter 8), it was concluded that if a signal's loss is worse than predicted, boundaries should attribute all of that loss. The reasoning behind this is that boundaries will naturally lose their signal modifiers where there is a possibility of a cell being responsible. Because a cell's *modifier* has such power when multiplied by the distance travelled through a cell, the *weights* of cells are almost always far higher than the weights of boundaries, even with such a generous *initial bonus*. Consequently, if the signal loss should be attributed to a cell, any signals internal to a cell will immediately bring the cell's *modifier* up, and its *weight* on future signals. And once a cell's *modifier* stabilises, the estimations become more and more accurate, meaning that the *ALD* will rarely differ from the *ELD*, and thus never give a boundary an opportunity to take any significant *initial bonuses* later on. The *initial bonus modifier*, *m*, is calculated as:

$$m = \frac{S_{ald} - S_{eld}}{B_{count}} \tag{4.8}$$

where S_{ald} is the signal's ALD, S_{eld} is the signal's ELD, and B_{count} be the number of boundaries applying for the *initial bonus*.

Finally, the new *modifier* value is calculated as the value with which to update the Averaging Window. This new *modifier* is what the communication object's *modifier* should be, according to this iteration of the algorithm. For cells, it is calculated as the *new logical distance* field divided by the *distance* a signal travels through the cell. For boundaries, it is simply the value of the *new logical distance* field. Table 4.2 on the next page summarises the *Update Modifiers Table* calculations.

Table 4.2: Update Modifiers Table

	ELD	**weight**	**new LD**	**new modifier**
cell	distance × modifier	ELD / signal's ELD	weight × signal's ALD	new LD / distance
boundary	modifier	ELD / signal's ELD	weight × signal's ALD **or** initial bonus	new LD

The *Update Modifiers Table* is sorted because of the special cases. If a cell's new *modifier* is artificially raised to 1.0, or if a boundary's *modifier* is 0 and may be given an *initial bonus*, then the currently calculated *weight*, *new logical distance*, and new *modifier* of each entry in the table need to be recalculated. If the table is sorted in ascending order, then boundaries with *modifiers* of 0, and cells with new *modifiers* below 1.0 will be at the beginning of the list. They will thus be calculated prior to other objects with much higher new *modifiers*. As each artificial adjustment is made, the object's *ELD* is subtracted from the signal's, and the adjusted *new LD* is subtracted from the signal's *ALD*. Thus, after the special cases have been calculated, regular entries in the *Update Modifiers Table* can be recalculated given the new signal *ELD* and *ALD* at that point. Their new *modifier* values will be lower than before, but these values are assumed to be more correct, as any adjustments made are aimed at improving accuracy.

The concept of boundaries adds an interesting alternative to the original Communication Map concept. The comparison of results between the two strategies is detailed in Chapter 8, and will show how both concepts fare in several differing scenarios.

4.6 Summary

A signal loss mapping solution has been developed to meet the criteria proposed for use in predicted routing (Section 4.2). This Communication Map meets these criteria through a number of means:

- Averaging Time Windows to average signal loss over short periods, using Time Blocks to average signal loss without storing each individual signal loss reading (Issue 1).
- The use of cells to represent areas of similar signal propagation (Issue 2 and 6).
- The proposed use of boundaries in addition to cells to more accurately represent signal propagation (Issue 2, 4 and 6).

- Map sharing among nodes to develop CM accuracy more rapidly (Issue 3).
- Algorithms for cell subdivisions to achieve varying levels of detail in real time (Issue 4).
- Default modifiers are assigned to new cells based on the signal loss averages of surrounding nodes (Issue 5).
- The proposal of a *modifier* value and the concept of *logical distance* to represent signal loss over multiple cells in relation to physical distance (Issue 7).
- The use of cell *weights* on an incoming signal reading in order to distribute signal loss more appropriately over cells. This way many cells may be developed without ever having a node physically travel within those cells (Issue 7).

Several limitations of the CM have been identified in this chapter. These issues should be investigated for real world implementations of this work. They have not been discussed as they are outside the technical scope of this thesis. They include:

- Analysis of the accuracy of GPS coordinate readings in relation to the CM.
- A study into the prevalence of signal loss amplification in natural environments.
- More accurately mapping signals across cells based on node position and movement relative to each cell.
- The implementation and study of different windowing techniques on the Averaging Window (such as [OS89], [LF98], [MF53]).
- A study into multi-path signal propagation.
- A survey of accuracy of different non-overlapping shapes for cells, as well as overlapping shapes, 3D models, and free-forming shapes.
- An investigation into alternatives to the *modifier* and *logical distance* concepts for multi-cell signal propagation effects.

The next chapter will look at using the CM to generate predicted connectivity information for use in wireless ad hoc routing protocols.

Chapter 5: Future Neighbours

The purpose of this research is to design a method for using signal loss maps and future node position information as a form of predictability in wireless ad hoc networks. The previous chapter detailed a solution for creating a signal loss map with wireless nodes. Such information can provide a basis for future node connectivity predictions when combined with future node mobility details. This list of future node connectivity predictions represents all nodes that will become neighbours of a node as far into the future as possible. This chapter presents such a solution, providing a list that can be immediately applied to routing algorithms to aid connectivity.

5.1 Motivation

Many routing algorithms have already been designed to efficiently implement routing in wireless ad hoc networks (e.g. [PB94], [JM96], [PC97], [KV98], [PR99], [LK00], [HJ01]). These algorithms have been designed to operate in various scenarios and take into consideration information from various sources. However, none of these algorithms or any wireless routing prediction algorithms found during this research use knowledge of future node mobility combined with a signal loss map. This information has the potential to improve connectivity in a number of routing protocols, investigated in Chapter 6.

In order to improve existing routing protocols as effortlessly as possible, this research aims to provide simple access to prediction information. The Future Neighbours concept is to provide a *table of future neighbours* for each node. This table lists the times when nodes in the network will become neighbours with each node, and the duration for which they will remain neighbours.

5.2 Issues that are raised

There are a number of issues related to developing a table of future node connectivity predictions. As with any form of predictability, there is primarily an issue of reliability. Routing protocols may have access to a table of future neighbouring nodes, but the accuracy of this table will determine how useful the information is, and to what extent

routing protocols can use it. Examples of where accuracy may be a factor are in the times at which connectivity is predicted to begin or terminate, or in the predicted connectivity as a whole, where a predicted connection may never arrive, or a predicted termination may never eventuate.

The delay of prediction updates is also a factor, as delays in information propagation to routing protocols may adversely affect connectivity. Depending on the activities of the network, each node's future tasks may change. More importantly, the signal loss map may also have a high rate of change, as areas are more accurately mapped with signal propagation statistics. The manner in which these changes propagate to connectivity predictions and are further propagated to routing algorithms becomes a factor.

As previously mentioned concerning algorithm efficiency in the CM, these algorithms are intended for wireless ad hoc nodes with potentially limited hardware capabilities. If network tasks and network environment changes are frequent, there is an even greater need for efficiency in these algorithms.

5.3 Design

To facilitate the use of future node location information, a Future Neighbours Table (FNT) is presented. This table simplifies future node mobility information by combining knowledge of future node locations (provided by neighbour Task Paths) with a CM of signal propagation. The result is a table of events in time when nodes will become or cease to be a neighbour with a node. This table can be immediately applied to existing routing protocols to improve network connectivity where future node movements are known. An example resulting FNT for two nodes is shown in Table 5.1 below.

Table 5.1: Example FNT

Time	Neighbours
0:00	true
1:42	false
2:57	true
8:20	false

A FNT is created for each pair of nodes in a network, regardless of their current connectivity status. As future node mobility and the CM are further developed and updated over time, so too must each FNT be recalculated at periodic intervals. A FNT is developed by estimating the signal loss between two nodes at the future locations those nodes are predicted to travel to. If this signal loss is within the maximum ASL range, then the two nodes are predicted to have connectivity (i.e. to be neighbours). When the signal loss falls outside the acceptable range, the two nodes are predicted not to be neighbours. The predicted signal loss is calculated through the use of the CM.

5.3.1 Task Paths

Future node movements are obtained from Task Paths. This thesis aims to improve routing primarily in automated wireless networks, where nodes act based on the set of instructions and tasks they have been assigned or will assign themselves. These Task Paths are propagated over the network to ensure every node connected to the network has knowledge of every other node's future predicted locations. In automated networks, changes to these Task Lists are shared with all nodes as they occur, so that future movement predictions are kept as accurate as possible. The FNT algorithms are given the Task Paths of two nodes as inputs.

Each Task Path consists of a list of {origin, destination} pairs which signify a direct path a node will take. Each origin and destination is made up of a pair of values, a time and a location, where the node will start and at what time, and where it will come to a halt or change direction, and at what time. This method has been chosen as it closely follows the travel paths that autonomous nodes are likely to be given in the fields this research has been focused on (e.g. [ABO05], [FT02], [OC96], and [THM95]). Autonomous nodes are often robots, which are commonly programmed to perform some task in an area (when they are not moving significantly), and then move on to other areas. When moving to other areas it is common for a robot to travel directly to the next location as this is the shortest path. Thus the Task Path design presented here has been modelled on the list of future tasks likely to be available to autonomous nodes. An example Task Path is shown in Table 5.2 on the next page. In this example a node will travel for 15 seconds in one direction, before changing directions and travelling for a further 10 seconds. Note that a node may stop between each travel line, depending on the tasks it is programmed to achieve.

Table 5.2: Example Task Path

Origin		Destination	
<Time>	**<Location>**	**<Time>**	**<Location>**
0:10	{1m, 25m}	0:25	{1m, 40m}
0:25	{1m, 40m}	0:35	{7m, 48m}

5.3.2 FNT Development

There are two ways to create a FNT that have been studied in this research. The first method is through polling (estimating signal loss at period intervals) and the second is through calculating the logical changes in the average signal loss modifier (which will be known as the linear method).

5.3.2.1 Polling Method

The easiest method to implement, although also often the slowest and least accurate, is polling. If the Task Paths of two nodes are known for some future period of time, at regular intervals the nodes' Task Paths can be 'polled' to find their predicted future location. By performing an estimated signal loss calculation via the CM given these two locations, the status of the two nodes being neighbours at that time can be determined. This process can be repeated to the extent that future predictions in time are desired. An example of the polling method is shown in Figure 23 on the following page. At regular intervals both of nodes A's and B's Task Paths are polled for future locations, and the signal loss is calculated at this interval.

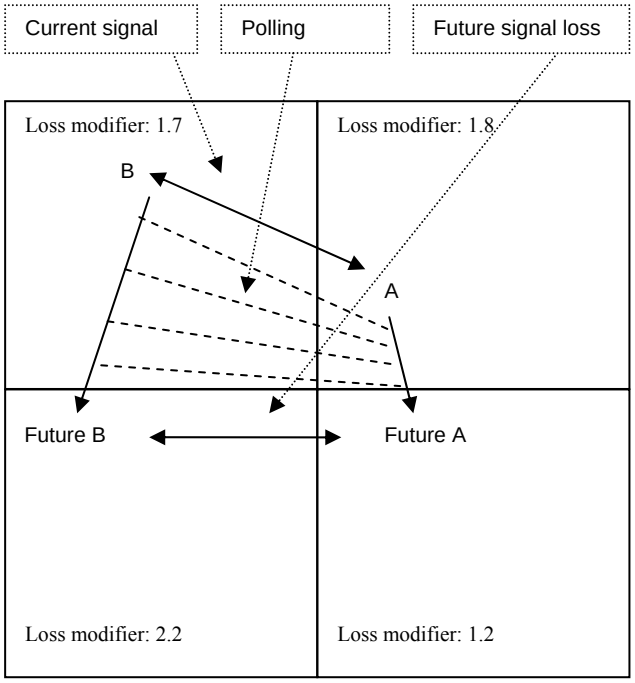

Figure 23: 'Polling' Example

Polling is slow because samples are taken at repeated intervals. Accuracy is a factor between these intervals, as no information is obtained during the periods between samples. Accuracy can be improved by reducing the period between intervals, but at a cost of speed.

5.3.2.2 Linear Method

The linear approach to generate a FNT is through calculating the logical changes in the average signal loss modifier. In most situations, given the limited maximum transmission range of wireless ad hoc devices (e.g. IEEE 802.11b standard network devices), and given the average size of cells in the CM, there are unlikely to be many cells between two nodes' Task Paths over time. For example, consider the maximum free-space loss range of Lucent Technologies Orinocco IEEE 802.11b interfaces of 478m (at the slowest transmission speed of 1mb/s having a receiver sensitivity of -94 dBm [Luc]). Given a MCS of 25m, the maximum number of cells between two tasks at any point is 20. To improve on the speed and accuracy over the polling method, a technique to find the points in time where a *change* in the average signal strength occurs would have a significant improvement in speed over polling at regular intervals, and with logically perfect accuracy.

As an example, in Figure 24, nodes A and B start with some value of signal loss between them. Because they are in a cell with an *average* signal loss modifier applied to distance, if the nodes move further apart at a linear rate, the signal loss will also increase at a linear rate. By finding the point in time where this signal loss is at a maximum, with regards to this cell alone, it is trivial to determine if both nodes are predicted to have connectivity during that time, or at what point in time they would become or cease to be communication neighbours. From this point on until a further point is reached, the signal loss will decrease linearly. As another cell's signal loss average comes into play, its starting minimum and finishing maximum signal loss can also be calculated, and added to the equation to produce a list of times and maximum / minimum signal losses. From this information it is computationally trivial to determine when nodes will and won't be neighbours over time. Thus, the linear method offers speed enhancements over the polling approach.

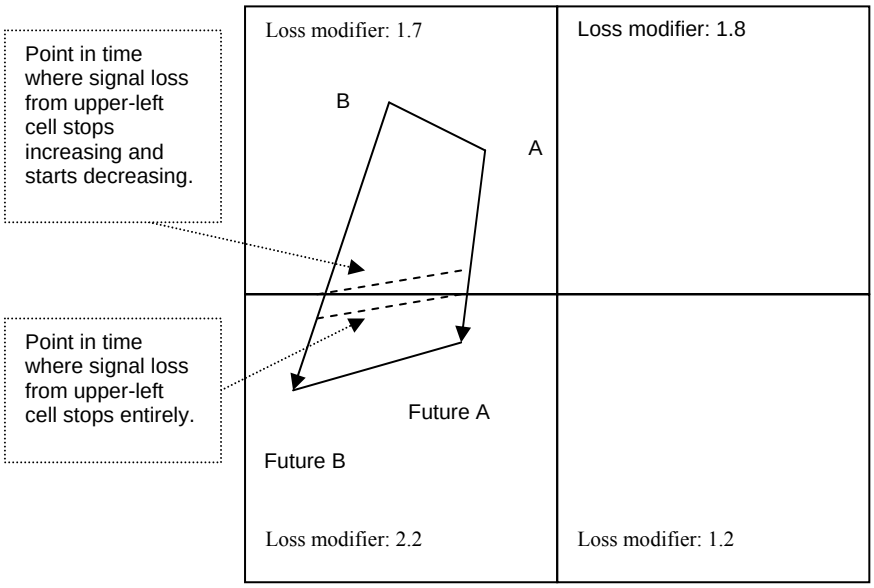

Figure 24: Average Signal Loss Change Example

There is a problem with the linear approach, however. After analysis of calculating the logical changes in the average signal loss modifier, it was discovered that not all situations are linear. In Figure 25, two nodes, E and F, start with significant distance between them. As they move, this distance reduces until such a point where the nodes have the least distance between them. From this point onwards, the nodes move further apart. If free-space loss is assumed, the signal loss will be parabolic. This is further complicated where cell signal loss modifiers are present. Figure 26 illustrates a scenario where there is a single simulated area with a high signal loss modifier. As nodes A and B travel, the

distance between them will both reduce and then increase at a non-linear rate. Because of the simulated signal loss area, as Node A travels in relation to Node B, it is constantly reducing the distance a signal will travel within the simulated signal loss area. This complicates the change in signal loss further, as there may be any number of periods of non-linear signal loss change over two task lines. Figure 27 graphs an example of this. For each cell with a differing signal loss modifier between two nodes, an added factor is introduced in calculating the logical distance between two tasks. Essentially, a polynomial of multiple turning points is possible for the rate of change of signal loss, limited by the maximum number of cells which may exist between nodes.

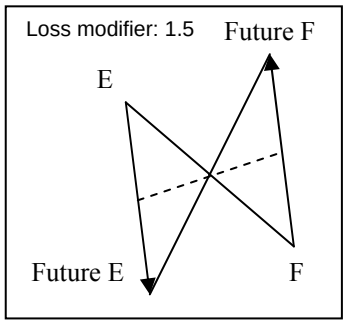

Figure 25: Parabolic Signal Loss

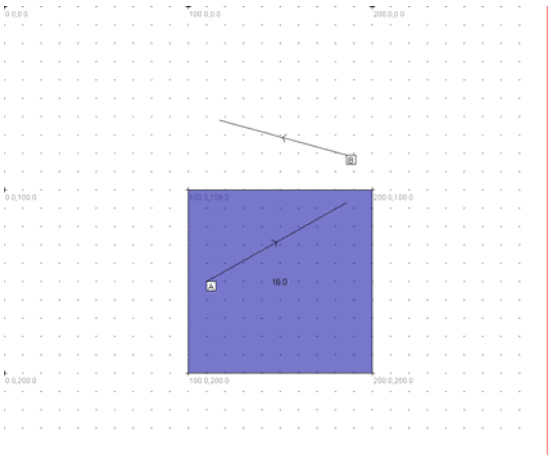

Figure 26: Simulated Scenario

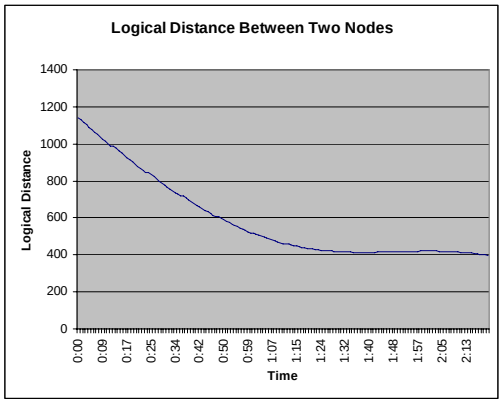

Figure 27: Logical Distance Between Two Nodes

For such cases, a simple mathematical solution for use in the linear method of creating FNTs has not been found. For any two task segments that experience this property it is necessary to use the polling method. There are two cases where it is possible to use the linear method. These two cases are based on whether both tasks are moving or if one is stationary. Two stationary tasks have no need for either method.

The first case exists where two moving tasks are travelling in directions that result in a linear change in distance between them. Refer to Figure 28 for this explanation. Two task lines exist, T_1 and T_2, and a virtual line connecting both task lines' starting points, T_3. Let T_{1s} and T_{2s} represent the starting points of two tasks, T_1 and T_2 respectively. Let T_{1d} and T_{2d} represent the destination points of T_1 and T_2 respectively. Let T_3 be the theoretical line between T_{1s} and T_{2s}. Let θT_1, θT_2, and θT_3 represent the bearings of T_1, T_2, and T_3 respectively. A linear change in distance between two nodes is true where:

$$\theta T_2 \in [\theta T_1, \theta T_3] \tag{5.1}$$

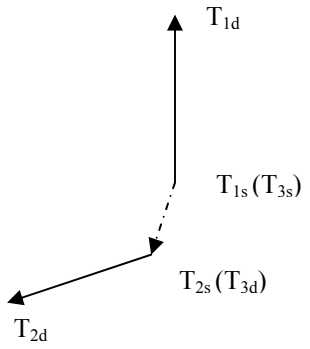

Figure 28: Direction of Task Travel

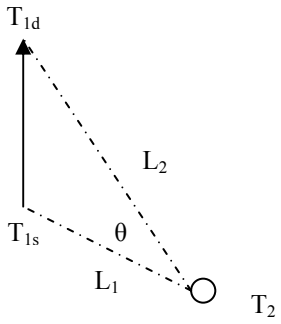

Figure 29: Angle between a Stationary Task and a Moving Task

The second case exists where a single moving task has a linear change in distance with that of a single stationary task. A linear change occurs so long as the distance between two tasks increases or decreases, but never both. If the relationship between a moving task's start and end points and a stationary task is viewed as a triangle, both moving task points (the start and end task points) must either be moving towards or moving away from the stationary task. This is illustrated in Figure 29. Given a moving task, T_1 and a stationary task, T_2, let T_{1s} and T_{1d} be the start and end points of T_1 respectively. Let L_1 and L_2 represent the theoretical line between T_2 and T_{1s}, and let L_2 represent the theoretical line between T_2 and T_{1d}. A linear change in distance between the two nodes occurs when:

$$\theta(L1,L2) < 90^\circ \tag{5.2}$$

Where the linear approach is usable, the most important variable to calculate is the *time* when some change occurs. There are two different cases where these time values will be calculated. The first case is illustrated back in Figure 24. If the beginning and end

locations of both lines are known, and both nodes start from the beginning and reach the end location at the same time, and if both nodes travel in a straight line at a constant speed, then finding the point in time where a change in *average signal strength* occurs can be found using a line intersection algorithm, where the boundary of a cell intersects both task lines. The second case is where a cell's boundary does not intersect either or both task lines. This is the case where a cell's corner resides inside the area that will be travelled around. This is demonstrated in Figure 30. A change occurs due to a change in cells between two nodes at some future point in time. In this situation the *point in time* is found where both nodes will find a cell's corner point precisely between their locations. To find this point, it is necessary to find the moment in time when both nodes' locations in time have an identical slope to the point between them.

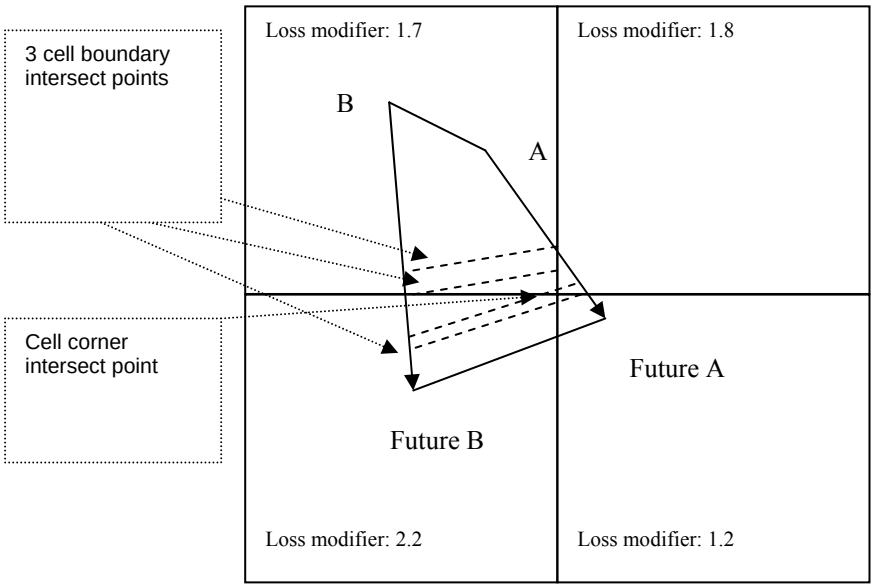

Figure 30: Example Point in Time

Finding these changes in the average signal loss modifier offers improved accuracy and speed over the polling method. However, the FNT's linear approach is algorithmically quite complex. Overall, two methods exist for creating a FNT. The linear approach will result in more accurate and often faster results over polling, but may only be used where the distance between two tasks changes at a constant linear rate. The polling method can be used in all other cases. Detailed comparisons between the two approaches are studied in Chapter 8. The algorithms to implement this will be detailed next.

5.4 Creating the Future Neighbours Table

Several algorithms must be performed in sequence to generate the FNT from a pair of task paths. This process is executed by each node for each other node in the network, provided one of the two nodes has future tasks. Each node therefore generates a list of future neighbours from all possible neighbours in the known network. The algorithms presented in this section are based on CM algorithms without boundaries. Boundaries are shown to have little accuracy benefits given their computational and bandwidth overhead (detailed later in Chapter 8).

Before the important points in time where a change in the average signal loss modifier can be found, the Task Path of each node must be processed. The Future Neighbours design relies on the Task Path having several properties:

- Only two, straight task lines are compared at a time;
- the start and end points of the line are known;
- both nodes leave their start point at the same time, and arrive at their destination point at the same time; and
- each node travels at a constant speed throughout the line.

The majority of the properties mentioned above are true based on the Task Path model chosen for this research. Nodes travel at a constant speed, and travel in straight lines by design. As Task Paths are known, all start and end points of each task line are also known. However, task start and end times vary for each node. Task Path *equalisation* is conducted whenever two task lines have differing start or end times. This is done to segment and align tasks such that they have these properties.

Following Task Path equalisation, creating a Signal Loss Over Time (SLOT) table, the main component of creating the FNT, is conducted. For every cell between two paired task lines, the times where a change in the average signal loss modifier occurs are recorded along with signal loss values. These times are obtained from either the linear or polling approach, based on the characteristics of the two task lines. For the linear method, these times are where a pair of tasks have a change in the cells that lay between them at any given time. A change in the intermediate cells means a possible change in the

signal loss modifiers being used, and consequently how a signal is predicted to propagate. With the polling approach, these times are where the signal loss rises above or falls below the maximum ASL. Finding these times forms the fundamental basis of the FNT concept.

The final FNT is generated by iterating over the list of times created in the SLOT table. Between each time in the SLOT table generated by the linear approach, the signal loss between two nodes in question will increase or decrease at a linear rate. The FNT is created by determining when the signal loss between two nodes is within acceptable levels or outside them, thus determining when two nodes will be communication neighbours over time. Figure 31 provides an overview of these three steps.

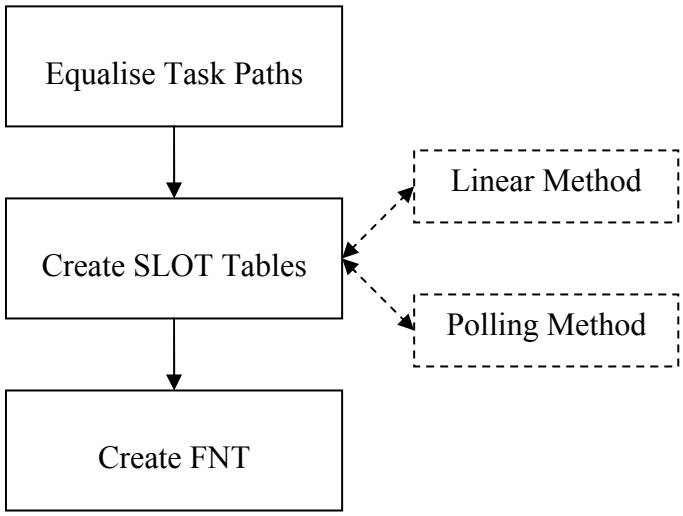

Figure 31: Overview of FNT Generation

5.4.1 Equalising Two Task Paths

Given a pair of Task Paths, the FNT first requires both Task Paths to be *equalised*. This stage generates two new Task Paths with matching beginning and end times for each pair of tasks. This process simplifies later algorithms in computing cell overlap changes. In order to compare any two future travelling nodes' tasks, both tasks must start and end at the same time. During a task, a node may either move at a constant speed or be stationary.

Algorithm 5.1 below is applied to two nodes' Task Paths, and segments both Task Paths into individual task lines that have identical start and end times. The algorithm iterates

through each pair of tasks from both Task Paths. There are three possible cases that may occur such that a change to one of the tasks is required:

Case 1: One of the two tasks could begin before the other task.
Case 2: One of the two tasks could end before the other task.
Case 3: One of the two task paths might have no tasks left to process.

In Case 1, where a task begins before its partner task, a stationary task is required in the partner's Task Path to match the start and end times of the earlier task. This fills in missing gaps in any Task Path where a node may be stationary. This scenario may be further complicated if the earlier task finishes after its partner Task Path begins its next task. Refer to Figure 32 below for an illustration of this situation. In this diagram, a stationary task is inserted to fill in the gap that exists in the later task's Task Path. However, the earlier task and the new stationary task still do not match. The earlier task must be segmented also if it finishes after the later task begins. This final segmentation is shown in Figure 33.

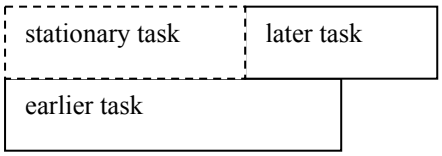

Figure 32: Preliminary Solution for Case 1

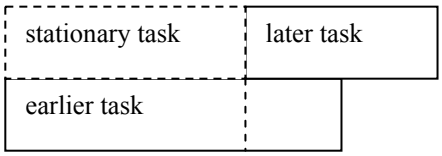

Figure 33: Final Solution for Case 1

Case 2 states that where two tasks do commence at the same time, there is the possibility that they will have differing end times (Figure 34). In such a situation the later ending task is segmented (Figure 35) such that its ending time matches that of the earlier task's ending time, and a new task is created following the later ending task. This task will then be the next task analysed in the Task Path of the associated node on the next iteration.

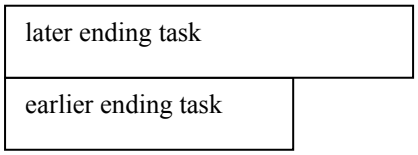

Figure 34: Case 2

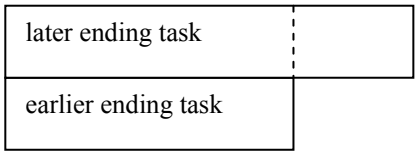

Figure 35: Solution for Case 2

And finally, Case 3 states that it is possible that one of two Task Paths have no more tasks to equalise (Figure 36). In such a situation a new stationary task is inserted for each remaining task in the partner Task Path (Figure 37). The origin and destination times are assigned identical times to the partner task, and the origin and destination location are set to the last location that the stationary node will stop at. Algorithm 5.1 on the next page shows the complete process of equalising two task paths.

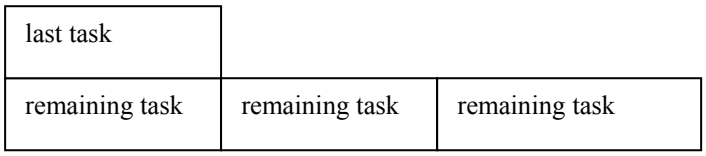

Figure 36: Case 3

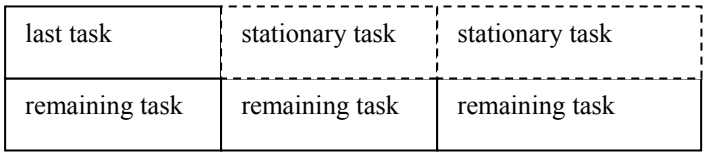

Figure 37: Solution for Case 3

Algorithm 5.1: Equalise Two Task Paths

- Let TP_a represent a copy of the Task Path of Node A.
- Let TP_b represent a copy of the Task Path of Node B.
- Let NT_a represent the next unprocessed task in TP_a, beginning at the first task.
- Let NT_b represent the next unprocessed task in TP_b, beginning at the first task.

1. Loop until the end of both task paths has been reached
   a. If both NT_a and NT_b are identical in origin and destination times, then move both NT_a and NT_b onto the next unprocessed task in TP_a and TP_b respectively.
   b. Otherwise if NT_a's origin time differs from NT_b's origin time, then:
      - Let $T_{earlier}$ represent either NT_a or NT_b, whichever has an earlier origin time, and let T_{later} represent the task with the later origin time, and TP_{later} represent either TP_a or TP_b, whichever contains T_{later}.
      - Let *earliestEndTime* represent either the destination time of $T_{earlier}$, or the origin time of T_{later}, whichever is earlier.

      i. Split $T_{earlier}$ into two, $T_{earlier1}$ and $T_{earlier2}$, by setting $T_{earlier1}$'s destination time to *earliestEndTime*, and creating a new task, $T_{earlier2}$, immediately following it, with an origin time of *earliestEndTime*, and a destination time of $T_{earlier}$'s original destination time.

      ii. Calculate and fill in the missing destination location for $T_{earlier1}$ and origin location for $T_{earlier2}$ given the new destination time of $T_{earlier1}$.

      iii. Add a task to TP_{later} such that its origin and destination times match those of $T_{earlier1}$, and its location for both origin and destination are identical to TP_{later}'s last task's destination location.
   c. Otherwise if both NT_a and NT_b both exist and have the same origin time:
      - Let $T_{earlier}$ represent either NT_a or NT_b, whichever has an earlier destination time,
      - Let T_{later} represent the task with the later destination time.

      i. Split T_{later} into two, $T_{later1\ and}\ T_{later2}$, by setting T_{later1}'s destination time to the destination time of $T_{earlier}$, and creating the new task, T_{later2}, immediately following it. Set T_{later2} with an origin time of the destination time of $T_{earlier}$, and a destination time of the original destination time of T_{later}.

      ii. Calculate and fill in the missing destination location for T_{later1} and origin location for T_{later2} given the new destination time of T_{later1}.
   d. Otherwise if only one task (either NT_a or NT_b) exists but the other node's task path has exhausted its supply of tasks, then:
      - Let $TP_{stationary}$ represent either TP_a or TP_b, whichever has finished processing all of its tasks.
      - Let T_{moving} represent the task (either NT_a or NT_b) which exists.

      i. Add a task to $TP_{stationary}$ such that its origin and destination times match those of T_{moving}, and its location for both origin and destination are identical to $TP_{stationary}$'s last task's destination location.
   e. Finally, move both NT_a and NT_b onto the next unprocessed task in the TP_a and TP_b lists, respectively.

5.4.2 Creating Signal Loss Over Time (SLOT) Tables using the Linear Method

The basis of the FNT design is to find the points in time where connectivity is predicted to be established or lost. Using the linear approach, this is where a change in the signal strength propagation rate occurs. The signal strength propagation rate is the linear rate at which signal loss in logical metres increases or decreases. With this information it becomes trivial to create a FNT that lists times when two nodes will gain and lose the property of being communication neighbours over time. A change in signal strength propagation rate occurs when, given any cell between two nodes at some future point in time, the rate of travel through a cell changes.

Two Task Paths, once equalised, contain a list of tasks with identical origin and destination times between both Task Paths. For every pair of matching tasks this then creates a physical triangle or quadrangle which defines the bounds within lie the signal propagation area which will affect the signal loss between two tasks. Figure 38 below shows the bounding shapes that may be created from two tasks. As seen from these examples, every pair of tasks communicate over one or more cells. This is true for all cases except two stationary tasks, where there are no calculations to perform and thus no need to create SLOT tables.

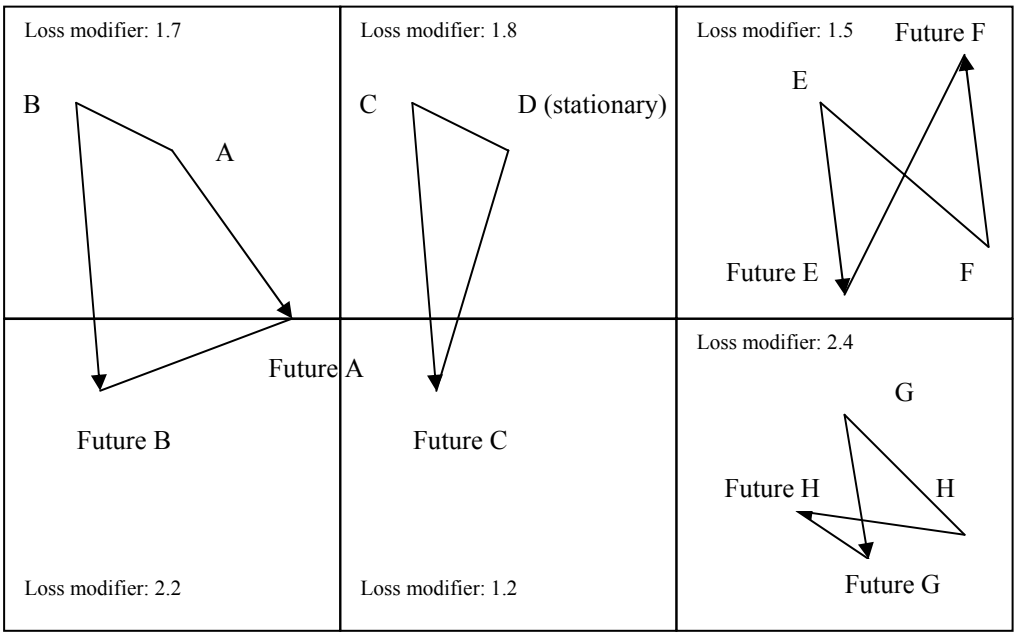

Figure 38: Possible Areas of Cell Overlap

Algorithm 5.2 creates SLOT tables for each cell that is predicted to affect the signal between two tasks. Each entry in this table is made up of two fields, *time* and *signal loss*. The time field indicates when a change in the passing through a cell occurs. The signal loss value is calculated from the assumed signal line that is created between the two task lines at the point in time indicated by the time field. The signal loss value is the overall signal loss between the two nodes at the future point in time found, not the signal loss in any given cell only. For each cell, points in time can be found in two types of instances:

1. Either of the nodes' task lines intersect one or two of the cell's boundaries.
2. Any of the four corner points of a cell lie within the bounds created from both nodes' task lines (within the shapes shown in Figure 38).

When a node's task line intersects a cell boundary, a change in the cells between two nodes occurs. Each cell may have a different signal loss modifier, and as new cells pass between the two nodes, the linear rate of signal loss change may be affected. When a node's task line intersects a cell boundary, such a change occurs. The point in time, t, where a node's task line intersects a cell's boundary can be calculated as:

$$t = \frac{d}{l_l}(l_e - l_s) + l_s \tag{5.3}$$

where d is the distance between the task's starting point and the intersection point with the cell's boundary, l_l is the distance between the task's starting and ending points, l_s is the starting time of the task, and l_e is the ending time of the task.

In many cases, nodes will travel either side of cells without intersecting cell boundaries. The signal loss modifiers of such cells come into effect when the cells are between the two nodes at some future point in time. The four corner points of these cells define when the signal loss *modifier* of these cells affects the signal between two nodes (Figure 39), and the rate of change that the distance of a signal passes through such cells. Given two tasks and a cell's corner point within the shape that represents the area they will communicate through, it is possible to calculate the point in time when the given point will be found between both nodes.

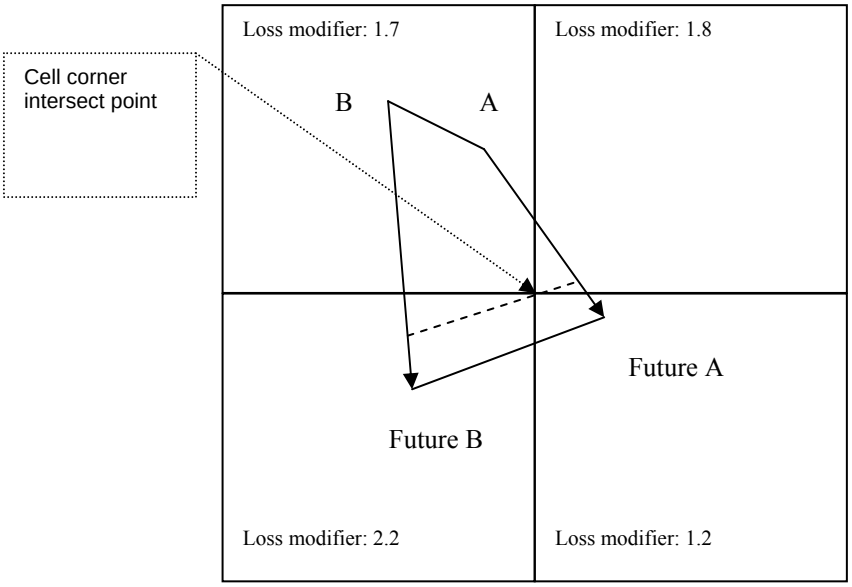

Figure 39: Example Point in Time

A point is found between two tasks when, effectively, both nodes find the point in a direct line of communication between them (even if it is only a logical point). Let (x_1, y_1) and (x_2, y_2) be the starting coordinates of the two tasks. Let (v_1^x, v_1^y) and (v_2^x, v_2^y) be the velocities of the two tasks in the x and y direction respectively. Also let d_1 and d_2 be the distances of tasks 1 and 2 respectively, and let (x_0, y_0) be the coordinates of the cell's corner point which lies between the two tasks. The point in time, t, can be calculated where the slopes between (x_0, y_0) and the position of each node along their respective task line is equal:

$$\frac{v_1^y \cdot t + y_1 - y_0}{v_1^x \cdot t + x_1 - x_0} = \frac{v_2^y \cdot t + y_2 - y_0}{v_2^x \cdot t + x_2 - x_0} \tag{5.4}$$

This can be solved for t , where $a \neq 0$, as:

$$t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{5.5}$$

This can be solved for t , where $a = 0$, as:

$$t = \frac{-c}{b} \tag{5.6}$$

where:

$$\mathrm{a} = (v_2^x \cdot v_1^y - v_1^x \cdot v_2^y)$$
$$\mathrm{b} = (v_2^x \cdot y_1 - v_2^x \cdot y_0 + x_2 \cdot v_1^y - x_0 \cdot v_1^y - v_1^x \cdot y_2 + v_1^x \cdot y_0 - x_1 \cdot v_2^y + x_0 \cdot v_2^y)$$
$$\mathrm{c} = x_2 \cdot y_1 - x_2 \cdot y_0 - x_0 \cdot y_1 - x_1 \cdot y_2 + x_1 \cdot y_0 + x_0 \cdot y_2$$

These two possibilities, boundary intersects and cell corner points, can be used to create SLOT tables using the linear approach. The algorithm to find these points in time where the cells between two nodes change is given in Algorithm 5.2 below.

Algorithm 5.2: Create SLOT Tables using Linear Method

- Let TL_1 and TL_2 represent the two task lines as input.
- Let TL_{start_time} and TL_{end_time} represent the start and end times of the two tasks.
- Calculate TL_{start_loss} and TL_{end_loss} as the signal loss between the two tasks at TL_{start_time} and TL_{end_time} respectively (using Algorithm 4.2).
- Let s represent the shape that is created from TL_1 and TL_2.
- Let l represent the list of cells which lie within the shape provided.
- Let t represent the SLOT table as an ordered table containing unique times.

1. Add { TL_{start_time}, TL_{start_loss} }to t.
2. Loop through each cell, c, in l:
   a. Iterate over each boundary, b, in c, where b intersects either of TL_1 or TL_2:
      - Let TL_x represent the task line, for each of the task lines (TL_1 and/or TL_2) that is intersected:
      i. Use Equation 5.3 to find TL_{time}, the point in time where TL_x intersects b.
      ii. Calculate TL_{loss} as the signal loss through this cell given the line between both task lines at TL_{time} (using Algorithm 4.2).
      iii. Add {TL_{time}, TL_{loss} }to t.
   b. Iterate over each point, p, of the cell, c:
      i. If p lies within s, calculate the point in time, p_{time}, when p is eclipsed by both nodes by solving Equation 5.4.
      ii. Calculate p_{loss} as the signal loss through this cell given the line between both task lines at p_{time} (using Algorithm 4.2).
      iii. Add {p_{time}, p_{loss} }to t.
3. Add { TL_{end_time}, TL_{end_loss} }to t.

As an example of this algorithm in operation, refer to Table 5.3 and Figure 40 on the following page. In this example, two nodes, A and B, are travelling along task paths where the start and end time are the same and speeds are constant (but may be different to each other). Using the top-left cell as an example, there are 3 points of interest that can be found. First is the origin time, as the signal line between both nodes will initially be passing through this cell. The signal loss here is given: 35 (logical metres). The second and third points of interest occur when one of the cell's boundaries intersects both task lines. The points in time when these occur are listed as at 25 and 32 seconds, and example signal losses of 50 and 60 are recorded. This table now contains the points in time and signal losses when changes in signal strength occur within a single cell. Because signal loss in logical meters has a linear property, between 0 and 25 seconds the signal loss increases at a linear rate from 35 to 50 logical meters, and then from 25 to 32 seconds, the signal loss through this cell increases more rapidly at a linear rate from 50 to 60 logical meters due to the next cell's higher signal loss modifier. From 32 seconds only the new cell's signal loss modifier is in effect, before finally the tasks end at 38 seconds. If there was a maximum acceptable signal loss of 55 logical meters, then the two nodes will have predicted signal connectivity until 28.5 seconds have elapsed. The alternate approach of developing SLOT tables using the polling method is presented next.

Table 5.3: Example SLOT Table

Time	Signal Loss
0:00	35
0:25	50
0:32	60
0:38	70

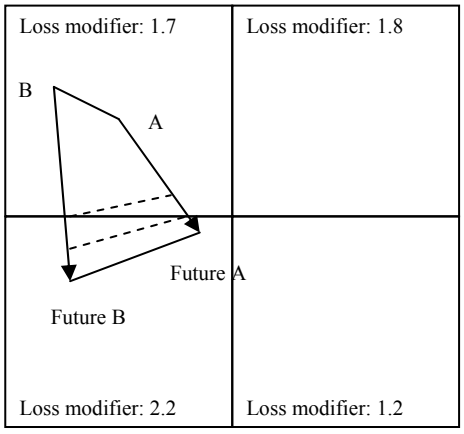

Figure 40: Example Cell Overlap

5.4.3 Creating Signal Loss Over Time (SLOT) Tables using the Polling Method

The polling approach to creating SLOT tables is algorithmically simpler than the linear approach, though computationally more expensive. At a regular polling interval, the predicted signal loss is calculated between the two tasks at each polled point in time. Signal loss predictions are obtained using the CM. Algorithm 5.3 describes this process.

Algorithm 5.3: Create SLOT Table using Polling Method

- Let TL_1 and TL_2 represent the two task lines as input.
- Let p represent the polling interval.
- Let t represent the list of cells which lie within the shape provided.
- Let s represent the SLOT table, as an ordered table containing unique times.

1. Set t to the start time of TL_1.
2. Repeat while t is less than the end time of TL_1:
   - Let s_{loss} represent the signal loss between TL_1 and TL_2 at the time t, calculated using Algorithm 4.2.
   a. Add t and s_{loss} as a new entry in s.

Both approaches to creating SLOT tables are used to generate a FNT based on whether signal loss predictions rise above or fall below the maximum ASL values (and any *safety margin* value used, discussed next). This final stage of developing the FNT is explained in the next section.

5.4.4 Creating the final Future Neighbours Table (FNT)

The final FNT is created based on the maximum Acceptable Signal Loss (ASL) configured for the network. A *safety margin* value may be introduced. A safety margin provides a buffer between the actual maximum ASL and the maximum ASL used for FNT predictions. Predicted connectivity is thus reduced where signal loss is too close to the maximum ASL. The safety margin reduces the maximum ASL value used in calculating FNTs. However, a greater safety margin value increases the likelihood of situations where connectivity is not predicted in a FNT but does exist between two nodes in the future. The safety margin values used should be based on the level of stability required in the network, and is discussed in Chapter 8.

Each entry in the SLOT table generated from Algorithm 5.2 contains a *time* and *signal loss* value. Where the signal loss between the current entry and the last entry fall on separate sides of the acceptable signal loss, the neighbour connectivity state changes. Finding the point in time, *t*, where the acceptable signal loss value is reached can be found by:

$$t = (\frac{m_{loss} - l_{loss}}{e_{loss} - l_{loss}})(e_{time} - l_{time}) + l_{time} \quad (5.7)$$

where m_{loss} represents the maximum acceptable signal loss setting, l_{loss} and l_{time} represent the signal loss and time of the last entry, and e_{loss} and e_{time} represent the signal loss and time of the current entry of the SLOT table.

Algorithm 5.4 on the following page shows the overall logic behind creating a Future Neighbours Table given two task paths.

Algorithm 5.4: Create FNT

- Let TP_1 and TP_2 represent the Task Paths of two nodes.
- Let f represent a FNT consisting of {time, neighbours} entries, where each entry specifies a time when the neighbour status between the two nodes changes, and to what the status is changed to.
- Let m_{Loss} represent the maximum ASL setting, minus the safety margin, that is acceptable for both nodes to consider themselves communication neighbours.
- Let n represent the status of the two nodes being neighbours at the current point in time.
- Let l_{loss} and l_{time} represent the signal loss and time of the last entry, where l_{loss} is initialised to the current actual signal loss between the two nodes in question, and l_{time} is initialised to the current system time that the nodes have recorded.
- Let s represent the SLOT table which will be created.

1. Equalise both task paths using Algorithm 5.1.
2. Iterate over the equalised Task Paths, TP_1 and TP_2, together. For each pair of tasks, $\{T_1, T_2\}$:
   a. Using Equations 5.1 and 5.2 on $\{T_1, T_2\}$, if the linear method may be used then:
      i. Update the SLOT table, s, for $\{T_1, T_2\}$ using Algorithm 5.2.
   b. Otherwise if the polling method must be used then:
      i. Update the SLOT table, s, for $\{T_1, T_2\}$ using Algorithm 5.3.
3. Iterate over s. For each entry, e, in s:
   a. If e_{loss} (the entry's signal loss value) is above m_{Loss} and n is true, then:
      i. set n to false
      ii. calculate the point in time, t, where this change occurs using Equation 5.7
      iii. record the values of n and t in a new entry in f.
   b. If e_{loss} is below m_{Loss} and n is false, then:
      i. set n to true
      ii. calculate the point in time, t, where this change occurs using Equation 5.7
      iii. record the values of n and t in a new entry in f.

5.5 Summary

The Future Neighbours Table is designed to provide a simple interface to the complexities of the Communication Map and tasking model. The resulting table is immediately applicable to ad hoc routing protocols, as it allows them to add neighbour predictions to improve connectivity. This table is generated for a pair of nodes in three steps:

1. Equalisation of both nodes' Task Paths.
2. Creation of SLOT tables.
3. Creation of the FNT for the pair of nodes.

The equalisation of Task Paths is required to ensure each task of one Task Path has a matching task (in terms of start and end times) in its neighbour's Task Path. The creation of SLOT tables uses two methods for creation: a linear and a polling approach. The linear approach is only available where there is a linear rate of change in distance between two nodes during a single pair of tasks. For non-linear rates of change, the polling approach is used. Finally, the final FNT is created based on the maximum ASL and any safety margin defined.

The algorithms presented in this chapter have some shortcomings. A FNT can only be generated where future node mobility is known. Also, the algorithms are designed only where nodes travel at a constant speed and in a straight line for each individual task.

The next chapter applies the CM and FNT algorithms to several existing routing protocols to analyse the benefits and weaknesses of the work of this thesis.

Chapter 6: Routing

The emphasis of this thesis is to design and implement prediction algorithms that can be immediately applied to existing routing protocols. Common among the routing protocols studied in this thesis is the use of a metric to select routes. This metric is often the distance (as a hop count) of the path between the source node and the destination node. The FNT can be applied to these routing protocols simply by replacing the existing metric with one based on predicted connectivity. Whereas existing protocols may use hop count, the FNT would provide the predicted duration of connectivity of routes. The connectivity of a route is based on the shortest period of connectivity between any pair of nodes along the route's path. Routing protocols employing the FNT algorithms would select routes with longer predicted connectivity over routes with shorter predicted connectivity.

Four routing protocols are studied in this chapter: two on-demand protocols and two distance-vector protocols. Both traditional protocols without prediction, and prediction protocols presented in Su's work, are studied. In addition, the modifications required to add the CM to Su's work, as well as the modifications to implement FNT in existing routing protocols, are provided. The CM is added to Su's work to determine how a signal loss map improves Su's work over the use of a free-space loss model. Although a greater number of traditional protocols could be studied, the modifications required to implement FNT are almost identical across a wide range of routing protocols. The FNT has been designed with this as a specific goal. For predicted connectivity-based protocols, only Su's algorithms are based on future node mobility, and as such are the only algorithms that can be compared with the FNT.

Both on-demand and distance-vector protocols are studied as the two primary categories of wireless ad hoc routing protocols. On-demand routing protocols create routes as nodes require connectivity. A comprehensive network state is not maintained, but rather individual connections across any number of nodes are formed as required. On-demand routing offers real-time route connections that suit applications with fast timing

requirements. However, on-demand protocols generically do not receive updates on the status of routes over time.

In contrast, distance-vector routing protocols receive regular updates of routes available through neighbouring nodes. Each node broadcasts a table of available routes to destination nodes, along with the metric used to select routes. Unlike on-demand protocols, however, when a connection breaks a new route is not found by distance vector protocols until routing updates are broadcast to potential source nodes. The routing protocols studied for on-demand routing are DSR [JM96] and FORP [Su00], and the distance-vector routing protocols studied are DSDV [PB94] and DV-MP [Su00]. Both FORP and DV-MP are based on Su's work, and use predicted connectivity as a routing metric.

6.1 Traditional Ad hoc Routing Protocols

6.1.1 Dynamic Source Routing

Dynamic Source Routing (DSR) [JM96] is an on-demand source-initiated routing protocol. Many on-demand protocols are based on the design of DSR (e.g. AODV, TORA), and thus the application of the FNT to it is significant. In DSR, each node is responsible for generating a route to any destination node through the route discovery process, if there does not exist a route in the node's route cache already. If a route exists in the cache with a valid unexpired timeout, that route is used to send the packet. If not, the source node broadcasts a route request to nearby neighbours. This Route Request (Figure 41) contains a source and destination address for the route, as well as a unique identification number (request ID) which identifies new information from stale information. The packet also contains a route record, which lists every node that forwards this packet. This forms a path between the source and the destination for the packet to traverse. The route record is also used as a metric, with the number of nodes signifying the distance between source and destination.

When the destination or some other node with a valid route to the destination is reached, a Route Reply is sent back with all the nodes along the path to the destination in the route record. If an intermediate node replies, then it will append the route record from

its cache to the current route record. This Route Reply packet has the same structure as the Route Request packet, shown in Figure 41.

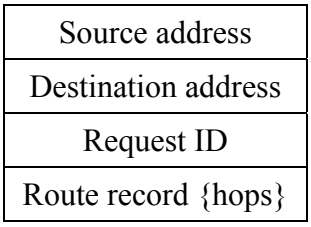

Figure 41: Route Request Packet Structure

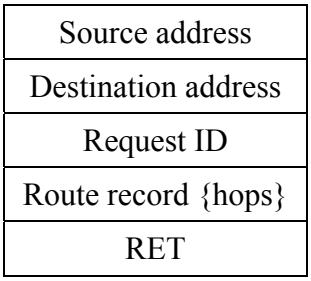

Figure 42: Modified Route Reply Packet Structure

DSR is studied as a traditional on-demand routing protocol. It uses distance as a metric and makes no use of prediction in any of its algorithms. DSR can be altered such that an additional field is transmitted in each Route Reply (Figure 42). A Route Expiry Time (RET) specifies the time at which the route between source and destination nodes is predicted to terminate. This value is determined by each node between the source and destination. Each node along the path which receives the Route Reply uses its own FNT to predict the expiry time of connectivity between itself and the node it received the Route Reply from (the previous node on the path). The new RET of the Route Reply is either the predicted expiry time between the previous node and the current node or the current RET, whichever is earlier. The resulting RET is the time when the first link between two nodes on the path between source and destination is predicted to fail.

Figure 43 illustrates an example of this. Given a path from Node D to Node A, the smallest expiry time between any two nodes is 3:20 (between nodes C and B), and is thus the expiry time of the entire route over this path of nodes.

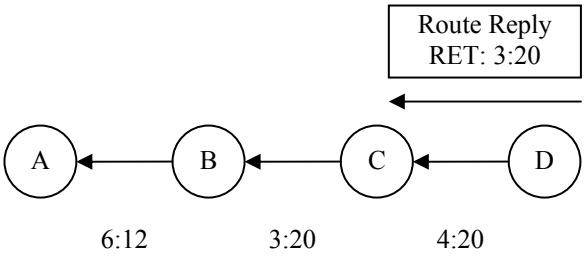

Figure 43: Example RET Calculation

6.1.2 Destination-Sequenced Distance Vector

Destination-Sequenced Distance Vector (DSDV) [PB94] is a single hop-based distance vector routing protocol. In DSDV, each node has a routing table which contains the details of the next hop for each destination it can reach, along with the number of hops to that destination and a sequence number (Figure 44 and Figure 45). Routing loops are avoided by using sequence numbers to distinguish stale routes from new ones. When a destination node sends out a broadcast, it assigns it a unique sequence number higher than the last, thus ensuring that receivers know this is new information from the destination. Any nodes receiving an update will use the highest sequence number. If two updates have the same sequence number, the route with the lowest hop count is used.

Hop counts are used as the route selection metric. As each node broadcasts a routing table, it increments each hop count value by one. This indicates that to use the path provided by the broadcasting node requires one more hop (the broadcasting node itself) to reach the destination than it takes from the broadcasting node.

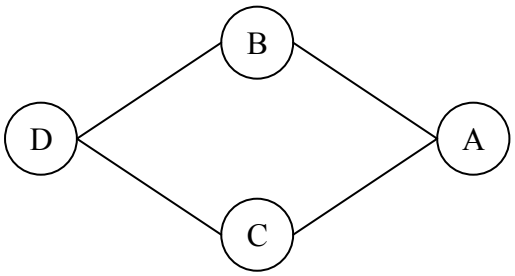

Figure 44: Example Node Connectivity

Destination	Next Hop	Hop Count	Sequence
A	n/a	n/a	3
B	B	1	2
C	C	1	2
D	B	2	2

Figure 45: Example Routing Table for Node A

DSDV can be modified to implement prediction using the FNT by replacing the hop count field in routing tables with RET values. As each node receives a routing table, it updates all RET fields. The new RET is either the existing RET value or the predicted expiry time between the node which broadcast the routing table and the node that received it, whichever is earlier. Each routing table therefore broadcasts a list of routes and the predicted duration of those routes. A node receiving several alternate paths via different nodes will then select the route predicted to remain viable the longest. This can further be used to dynamically alter the frequency of routing broadcasts, although such research is left for future investigation.

6.1.3 Applying the FNT to Routing Protocols

The FNT has been designed to provide an alternate or additional metric to existing routing protocols. As such, given any pair of nodes, a table of periods of connectivity can be computed. More advanced routing protocols could be developed to fully use this table, including future periods of connectivity where none are currently established. However such designs are not the purpose of this thesis, and consequentially only RET values, similar in design to concepts in [Su00], are used.

The FNT is calculated at periodic intervals or at a change in future Task Paths of a node. The FNT is not recalculated at changes to the CM, as these changes are too frequent. As routing protocols require a route metric, the FNT calculates the expiry time of connectivity between a pair of nodes along the routing path. This is presented in Algorithm 6.1 on the following page.

Algorithm 6.1: Calculate Predicted Expiry Time

- Let *s* be the source node with which connectivity is being predicted.
- Let *d* be the current node to which *s* is neighbours with.
- Let *t* be the FNT between *s* and *d*, where *t[i].time* represents the time of entry *i* in the table, and where *t[i].neighbours* represents whether or not the two nodes, *s* and *d*, are predicted to be neighbours at *t[i].time*
- Let *PET* be the Predicted Expiry Time that this algorithm will calculate.
- Let *c* be the current time (in seconds).
- Let *neighbours{s, d}* be a table of current connectivity status between pairs of nodes.

1. If the size of *t* is 0:
   a. Assign *PET* the value of *infinity*.
2. Otherwise:
   - Let *p* be the previous entry in *t*.
   a. If *c* is less than *t[first entry]* then:
      i. If *neighbours{s ,d}* is true, assign *PET* the value of *t[first entry].time*.
      ii. Otherwise assign *PET* the value of *-1*.
   b. Iterate over each entry, *e*, of *t*:
      i. If *c* < *t[e].time*, then:
         I. If *t[p].neighbours* is *true*, assign *PET* the value of *t[e].time*.
         II. Otherwise, assign *PET* the value of *-1*.
      ii. Assign *p* the value of *e*
   c. If *c* > *t[last entry].time*, then:
      i. If *t[last entry].neighbours* is *true*, assign *PET* the value of *infinity*
      ii. Otherwise, assign *PET* the value of *-1*.

Algorithm 6.1 calculates the predicted expiry time between two nodes, *s* and *d*, by finding the relevant entry in the FNT which predicts this connectivity. The value is obtained from the FNT entry which predicts connectivity to break after the present time (value *c*). If there are no predictions of connectivity after the current time, or if the FNT contains no entries at all, then the predicted expiry time is given a value of infinity, indicating that the link is never predicted to expire. If the two nodes are not currently predicted to be neighbours, then the predicted expiry time is set to -1. This can occur if the CM predicts signal loss in an area higher than actually exists. By finding these predicted expiry times, the FNT can be applied to existing wireless ad hoc routing protocols with ease.

6.2 Su's Algorithms

6.2.1 Flow Oriented Routing Protocol

Su [Su00] proposed the Flow Oriented Routing Protocol (FORP), an on-demand routing protocol using predicted connectivity as a routing metric. Based on existing on-demand routing protocols (DSR, AODV, TORA, etc), routes are established and maintained only where they are required.

Route requests in FORP operate similarly to DSR. A Flow-REQ packet (refer to Figure 46) is broadcast from a sending node requiring a new connection. Each node receiving this packet broadcasts it until the destination is reached. Sequence numbers ensure that messages are broadcast only when a higher sequence numbered packet is received than has already been seen, or where the same sequence number has a better path.

Source ID
Destination ID
Sequence Number
Path: list of {Node ID, LET} pairs

Figure 46: Flow-REQ Packet Structure

FORP has been designed with route stability as its primary goal. Path quality is determined by routes with the longest predicted connection time. This is in contrast to existing algorithms (e.g. DSR, AODV, TORA) which focus on shortest-path as the route selection mechanism. Path selection is also based on RET values, the time at which a route is predicted to terminate. The RET value is the smallest Link Expiry Time (LET) value of all pairs of nodes in the path between source and destination nodes (refer to Section 6.2.3 for a detailed explanation of LET values). Each node, as it broadcasts the Flow-REQ packet, adds its own node ID and LET values to the end of the packet, detailing the path over which this Flow-REQ packet travels. All nodes are assumed to have identical time references (provided by GPS, for example). As Flow-REQ packets are received by the destination, the path with the greatest RET is selected, and a Flow-SETUP packet is returned along the selected path back to the source node.

As FORP already uses a prediction metric, applying the FNT is relatively simple. LET values are replaced with values calculated using Algorithm 6.1 presented earlier, the remainder of FORP can remain unchanged.

6.2.2 Distance-Vector Protocol with Mobility Prediction

[Su00] also proposed a distance vector protocol based on his mobility prediction design. The Distance-Vector Protocol with Mobility Prediction (DV-MP) is based on existing distance vector protocols such as DSDV and WRP [MG95]. Each node maintains a routing table similar to that in DSDV, containing destination, next hop and sequence number. DV-MP's purpose is to increase route longevity, the same as with FORP. As such, RET values are recorded and routes are selected based on the highest RET values. These RET values are the same as discussed for FORP. The RET is calculated such that the RET is equal to the transmitted RET in a routing table update packet, or the LET between the receiving node and the broadcasting node of the routing table update packet, whichever is smaller.

An example is shown in Figure 47 and Figure 48. Four nodes are connected together with LET values as indicated. Each node broadcasts their routing tables at periodic intervals. Sequence numbers are assigned for each route only by the destination node itself. This ensures only one node for any destination is incrementing the sequence numbers. The routing table in Figure 48 shows that Node A has direct links to nodes B and C, and uses Node C as the next hop to Node D. The path A-C-D has a greater RET (of 3) than the path A-B-D (which has a RET of 2). This implies that path A-C-D is predicted to remain connected longer than path A-B-D. Real RET values are based on standard time, such as that available from GPS satellites.

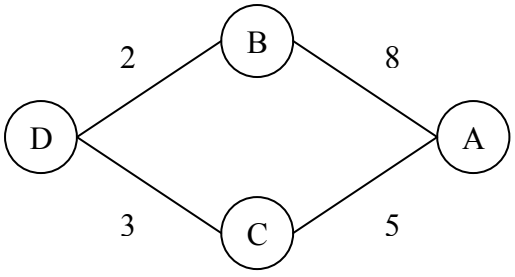

Figure 47: Example Node Connectivity

Destination	Next Hop	RET	Sequence
A	n/a	n/a	3
B	B	8	2
C	C	5	2
D	C	3	2

Figure 48: Example Routing Table for Node A

Applying the FNT to DV-MP is similar to that of FORP, replacing the existing LET values with those calculated using Algorithm 6.1. The routing table structure is identical to that described earlier in Section 6.1.2 on the modifications to DSDV required for use with the FNT. The use of a prediction metric is based on the use of the majority of metrics in routing protocols, where a single value is used to represent the preference of routes.

6.2.3 LET Predictions

FORP and DV-MP base LET prediction values on each node's immediate direction and a free-space propagation model. The complete task path of nodes is not considered in Su's model, as in Su's work there is no assumption of future node movements being made available [Su00]. Thus unlike the FNT, LET predictions use only the current speed and direction of travel to calculate when the connectivity between two nodes is predicted to terminate.

Let TX be the maximum transmission range over the wireless interface and (x_1,y_1) and (x_2,y_2) be the coordinates for nodes 1 and 2, respectively. Let V_1 and V_2 be the speeds of nodes 1 and 2, θ_1 and θ_2 be the headings of nodes 1 and 2, respectively. The formula used to calculate D_t, the amount of time that two nodes will remain connected, is given by:

$$D_t = \frac{-(ab+cd)+\sqrt{(a^2+c^2)TX^2-(ad-cb)^2}}{(a^2+c^2)} \qquad (6.1)$$

where

$$a = V_1 \cos\theta_1 - V_2 \cos\theta_2$$
$$b = x_1 - x_2$$
$$c = V_1 \sin\theta_1 - V_2 \sin\theta_2$$
$$d = y_1 - y_2$$

This formula from Su's algorithms [Su00] contains the flaw identified in Section 5.3.2.2, where the rate of signal loss change is not guaranteed to be linear. With a varying rate of change, it becomes possible to have multiple periods of connectivity from a single pair of nodes moving in straight lines at a constant speed. As such, testing of Su's algorithms in Chapter 8 is modified to use the FNT algorithms given only the speed and direction of travel for a pair of nodes, and without implementing the CM. This produces the same results as Su's LET prediction formula (Equation 6.1) for linear cases, and provides accurate LETs where non-linear cases occur. This is discussed further in Section 6.2.4.

6.2.4 Modifying Su's Algorithms to use the CM

[Su00] uses a free-space signal propagation model for all connectivity predictions. Free-space signal loss is simple to calculate, but rarely found in real-world environments [FTL95][HP02][HSS03]. This thesis proposes a modification to both FORP and DV-MP to use the CM instead of the free-space signal loss model. It is thereby possible to compare the CM specifically in relation to using the "no signal loss" map in Su's work.

Due to the flaw of non-linear cases found in Su's work, the implementations of FORP and DV-MP in Chapters 7 and 8 use a modified version of the FNT algorithms based on Su's approach. LET predictions are made using the FNT but without using the CM as follows. Wherever a prediction of the signal loss between two points is required the free-space loss model is applied. The FNT algorithms are given only the speed and direction of travel in which to base predictions on, as opposed to a full task list as designed. This approach simulates Equation 6.1 earlier, producing identical results for linear cases and accurate results for non-linear cases where the polling method is employed. Algorithm 6.2 details how LET values are calculated for Su's protocols based on FNT. Note that Algorithms 5.2 and 5.3 use a free-space signal loss model when estimating signal loss via Algorithm 4.2, instead of using the CM. Let θn be the bearing of direction of a node, and n_{speed} as the speed of that node (in metres per second). Let D_{max} be the maximum distance that a signal can travel in free space without natural amplification given the maximum

ASL, using Equation 4.4 (Section 4.4.1). Two artificial task lines are created given the location of both nodes, along with their speed and direction of travel. A task line, t_{line}, is defined as:

$$t_{line} = (s_x, s_y, t_{start})\text{-}(d_x, d_y, t_{end}) \quad (6.2)$$

Where s_x and s_y are the starting location's x and y coordinates, d_x and d_y are the node's destination locations, and t_{start} and t_{end} are the task's starting and ending times. We define t_{start} to be the current time, and t_{end} as:

$$t_{end} = t_{start} + n_{speed} \,.\, D_{max} \quad (6.3)$$

We define d_x and d_y as:

$$d_x = \cos(\theta n) \,.\, n_{speed} \,.\, D_{max} \quad (6.4)$$
$$d_y = \sin(\theta n) \,.\, n_{speed} \,.\, D_{max} \quad (6.5)$$

Algorithm 6.2: Modified Calculate Link Expiry Time

- Let T_1, and T_2 represent the two task lines of two given nodes, created using Equation 6.2
- Let m_{Loss} represent the maximum ASL setting, minus the safety margin, that is acceptable for both nodes to consider themselves communication neighbours.
- Let n represent the status of the two nodes being neighbours at the current point in time.
- Let s represent the SLOT table which will be created.
- Let *LET* represent the Link Expiry Time which this algorithm will calculate.

1. Using Equations 5.1 and 5.2 on $\{T_1, T_2\}$, if the linear method may be used then:
   a. Update the SLOT table, s, for $\{T_1, T_2\}$ using Algorithm 5.2.
2. Otherwise if the polling method must be used then:
   a. Update the SLOT table, s, for $\{T_1, T_2\}$ using Algorithm 5.3.
3. Iterate over s while LET is undefined. For each entry, e, in s:
   a. If e_{loss} (the entry's signal loss value) is above m_{Loss} and n is true, then:
      i. calculate *LET* as the point in time where this change occurs using Equation 5.7
4. If *LET* is undefined:
   a. Assign *LET* the value of T_1's end time.

To experiment with the CM in relation to Su's algorithms, alternate versions of FORP and DV-MP do not use Algorithm 6.2 but rather use the CM for predictions, as the FNT algorithms were originally designed to be used. These are discussed later in Chapter 8.

6.3 Summary

The routing protocols presented in this chapter are all easily modified to make use of the FNT. The FNT was designed specifically to provide a layer of abstraction between routing protocols and the details of generating connectivity predictions between wireless ad hoc nodes. The aim of the information that the FNT provides is to improve route stability and connectivity over longer durations. Similarly, FORP and DV-MP have been designed for the same motivation, and will be analysed in relation to FNT algorithms. FORP and DV-MP are also modified such that the accuracy of using the CM combined with them can be studied.

The routing protocols DSR and DSDV have been presented as the two most common foundation routing protocols used in wireless ad hoc networks. Numerous ad hoc routing protocols originate from DSR and DSDV for use in wireless networks (including FORP and DV-MP). The ability to apply the FNT to these protocols demonstrates the type of modifications to existing protocols required. It also identifies the potential for performance improvement in route stability, as will be detailed in Chapter 8. The following chapters will present the simulation tool developed for testing these algorithms, as well as the test results themselves.

Chapter 7: Simulation

7.1 Wireless Ad hoc Network Simulator (WANS)

The previous chapters have introduced novel concepts in predicting route expiry times. This has been done to improve routing connectivity in wireless ad hoc networks, regardless of the routing algorithm used. Simulations are required to validate these concepts, while identifying various strengths and weaknesses among algorithms and various settings.

The aim of this chapter is to describe the custom simulator tool used to test the various components of the algorithms described in earlier chapters. Simulators ([Fal99], [Mcd91], [UAG94], [Kes88]) already exist for basic network simulation. While many of these simulators are extensible, none specifically addresses the issues of wireless signal mapping and signal loss map testing. This is a significant issue to the core design of a simulator as simulated signal propagation characteristics are vital for testing. It is because of this that a custom simulator was created by the author of this thesis, the Wireless Ad hoc Network Simulator (WANS). This simulator was created to provide an appropriate testing bed for wireless ad hoc protocols that rely specifically on signal loss maps. The design goals of this simulator are to:

- visually create networking environments;
- run simulations in real or simulated time;
- visually inspect the accuracy of the network and its performance;
- obtain a variety of statistics;
- save and load simulations for later re-use; and
- batch execute a group of settings over a group of simulations.

7.1.1 Overview

In order to understand the results presented in this chapter, the details of the simulator are presented. A description of the simulator's implementation is available in Appendix A.

Figure 49 shows the core components of the simulator along with the primary flows of information. At the base of all algorithms is the node, where the network as a whole consists of any number of simulated nodes which execute the various wireless protocols being tested. Control of nodes and node movement falls under the role of the node simulator component. The CM, FNT, and various routing protocols all execute for each node in the network. Each node also maintains the Current Neighbours component, to monitor node connectivity with other nodes in the network. Each node has a network connection which is used to transmit and receive packets through the simulated wireless network. The network connection also relies on information provided by the routing protocol where a packet is directed to a specific destination. Statistics and snapshots of each simulation are collected from each node.

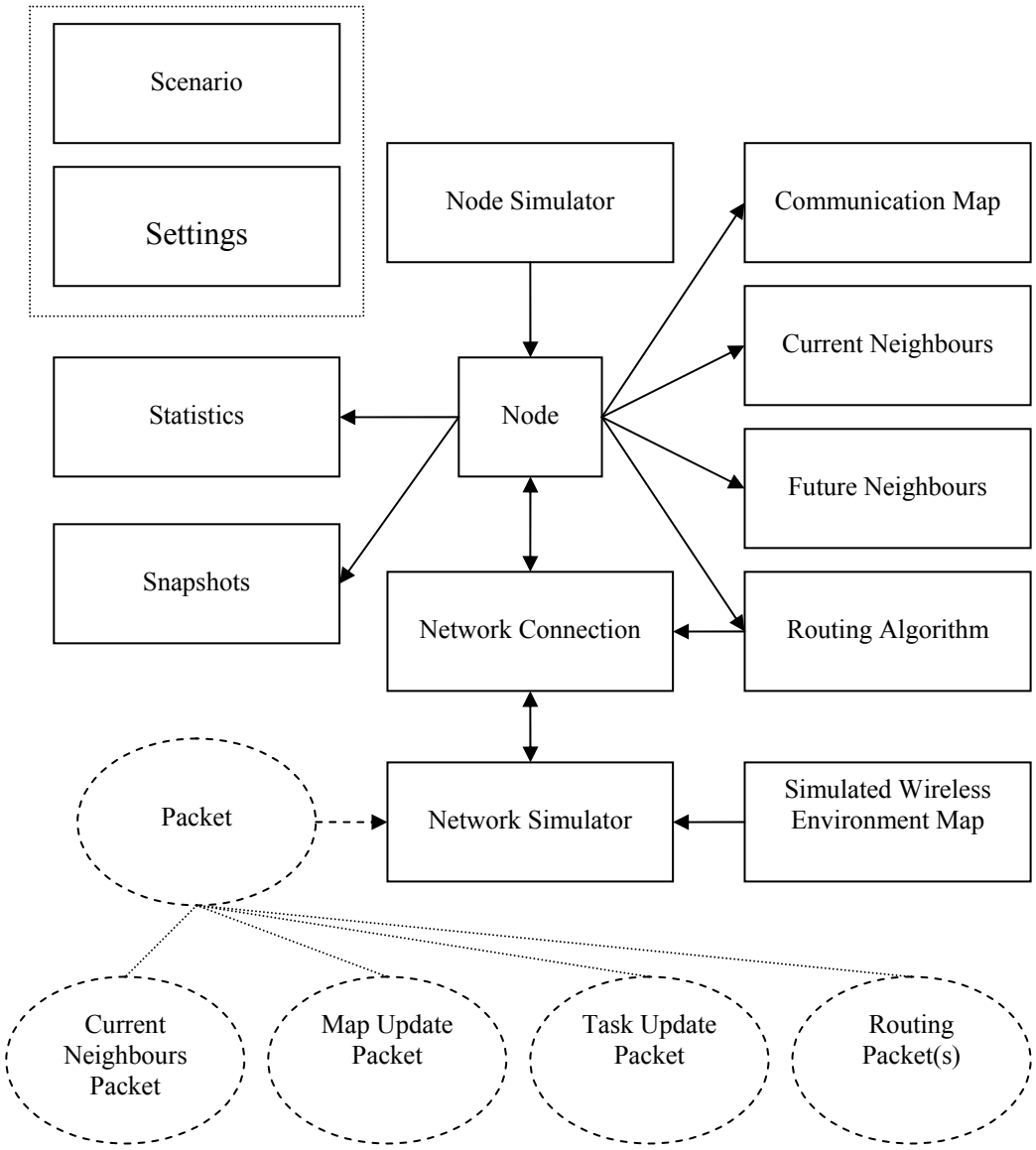

Figure 49: Simulator Overview

Each simulation has two inputs, a scenario file and a settings file. Each scenario details node positioning and movement, and the simulated wireless propagation environment. The settings describe the various attributes of each algorithm and module that are variable. In this fashion, a particular group of settings may be applied to multiple scenarios and a particular scenario may be executed using multiple groups of settings. The simulator is fully parameterised from these two inputs. An example scenario, showing several nodes, their task paths, and regions of different signal loss is shown in Figure 50.

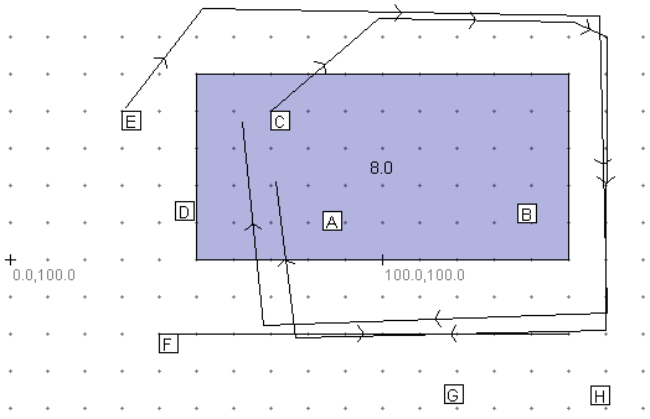

Figure 50: Example Scenario

7.1.2 Statistics

Statistics are collected to analyse the performance of various algorithms, settings, and scenarios. This data is periodically collected and summarised based on user settings. Each node in the simulator collects information for each module or algorithm that it implements. The periodic collection collates this information together to generate overall statistics. This periodic collection can be performed at any interval, depending on the level of detail required. For most simulations performed in the next chapter, this collection is performed once per second. The statistics collected record the minimum, maximum and average values for each set of data collected. The sets collected are:

- Set 1: Communication Map error (% error margin);
- Set 2: Communication Map differences between nodes (% error margin);
- Set 3: Future Neighbours Table predicted signal loss error (% error margin);
- Set 4: Future Neighbours Table accuracy (%);

- Set 5: Future Neighbours Table linear / polling time taken (seconds);
- Set 6: Future Neighbours Table linear / polling ratio of cases (%);
- Set 7: Routing hop count;
- Set 8: Routing unexpected drop-off count;
- Set 9: Routing time without route due to drop-off;
- Set 10: Routing time without connectivity (including drop-off period);
- Set 11: Routing predicted connectivity error (%); and
- Set 12: Bandwidth based on packet type over the simulation.

A number of these sets measure the error margin, which is the difference between the actual situation and each algorithm's understanding of the situation. This is measured as a percentage. An increase in error margin percentage indicates poorer performance. Set 1 measures the difference between the CM's estimated signal loss and the actual signal loss reported by the network simulator for each packet that is received. Set 2 compares the average modifier of each cell with the average of the average modifier for that cell over the CMs of all other nodes. Set 3 measures the difference between the signal loss between each node and the predicted signal loss provided by FNT algorithms. Set 4 expands on Set 3, but details how accurate the connectivity predictions were between nodes. As such, an accuracy of 100% indicates all predictions of connectivity or lack of connectivity were true. The amount of time during each interval that a prediction is false will adversely affect this accuracy. Sets 5 and 6 analyse performance between the linear and polling approaches to FNT generation. Set 5 specifically looks at the time taken for each approach, while Set 6 identifies the number of cases where the linear approach could be applied in relation to the number of cases where the polling approach was required.

Sets 7 through 11 analyse routing performance. Routing performance is ultimately how the strengths and weaknesses of all algorithms presented in this research will be tested, as improving connectivity in wireless ad hoc networks is the primary aim. Set 7 collects information on the average number of hops to all destinations from each node. Set 8 records the average number of connections that were unexpectedly terminated due to a loss of connectivity somewhere in the path to a destination. Set 9 and 10 collect the average percentage of time nodes spend without a connection to a destination, with Set 9measuring this only where it resulted from an unexpected drop-off, and Set 10

measuring the time without connectivity in total. Set 11 applies only to routing protocols implementing some form of prediction. It records the percentage of time that a path to a destination was incorrectly predicted to have connectivity.

Set 12 is included to analyse the amount of bandwidth consumed for each of the various protocols tested. For all sets, the second phase of statistics outputs tabulated results for the entire scenario at the end of each simulation. This information is saved as multiple comma-separated value files for further user analysis or script parsing.

7.1.3 Snapshots

Snapshots give a detailed view of the network and the various algorithms in operation. This is in contrast to statistics, where information is used to compare the overall accuracy of algorithms, settings and scenarios. Snapshots are provided at any required interval, and generate a number of views of the network and its topology at a given time. Each of the views is either a graphical image, a set of images or a table of data. The views available are:

- View 1: Simulated wireless map overlayed by current node movement combined with future task paths.
- View 2: Communication Map of each node;
- View 3: Connectivity between nodes;
- View 4: Connectivity clusters between nodes;
- View 5: The Task Path of each node;
- View 6: FNT cell overlap of each node for each equalised task;
- View 7: The FNT for each node with every other node;
- View 8: Node neighbour history;
- View 9: Route history of each node; and
- View 10: Current routing table of each node.

View 1 is presented to identify current and future node movement in relation to the simulated wireless map. View 2 visually details how each node represents the signal loss found in the physical environment. Figure 51 shows a CM for Node A at 10 seconds into a simulation. The node positions at this time as well as node A's CM are represented.

View 3 provides a simple view of the effects of a simulated wireless map in relation to the distance between nodes (see Figure 52 as an example). Similarly, View 4 illustrates each group of nodes which have connectivity with each other node in that group (see Figure 53 as an example of this). The tasks that each node is presently executing and have plotted into the future is available in View 5 (example shown in Figure 54). Figure 55 shows an example of View 6, illustrating the important points in time as found by the FNT algorithms between the stationary Node A and the related segment of Node F's Task Path. The beginning and end points of this task are drawn, as well as a cell point intersection and a boundary intersection between the two lower cells. There are four cells in total shown to have an effect on the signal loss over the illustrated task segment. Logical distance values are labelled beside each important signal line between the two nodes.

View 7 presents the current FNT of each node in the network, based on the last FNT generation performed. View 8 lists the history of connectivity between each node in the network, required when analysing FNT prediction accuracy. View 9 and 10 present routing tables, with View 9 representing the history of routes, and View 10 representing the current routing table. As a route is maintained where possible between all nodes in the network, routing tables across different routing protocols can be compared in fairness.

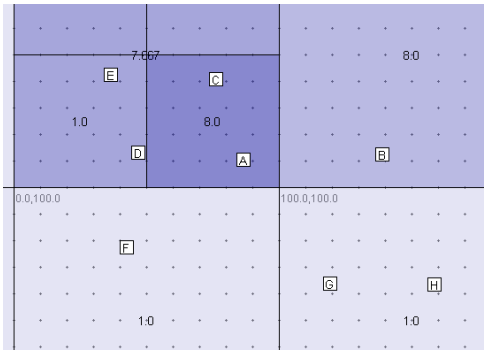

Figure 51: Example Communication Map

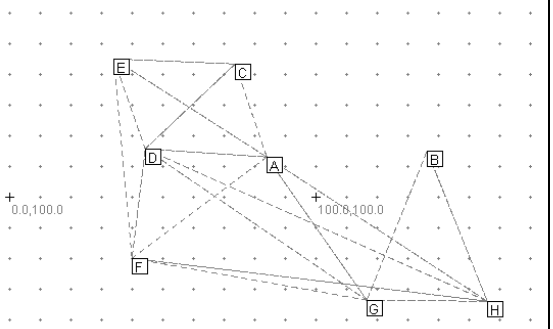

Figure 52: Example Connectivity Between Nodes

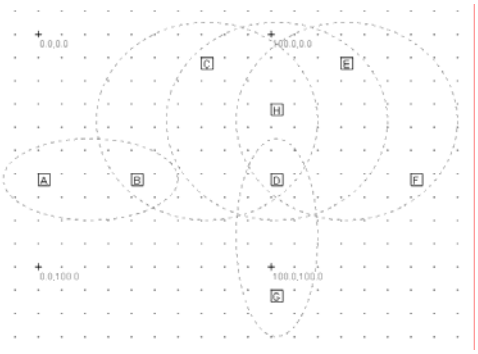

Figure 53: Example Clustering

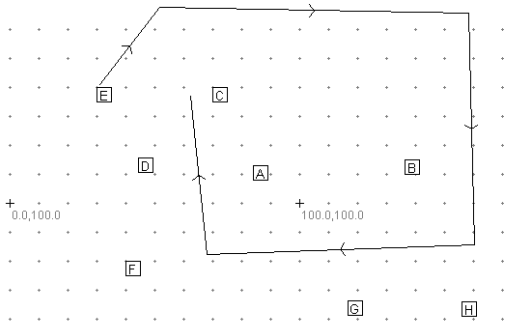

Figure 54: Example Task Path

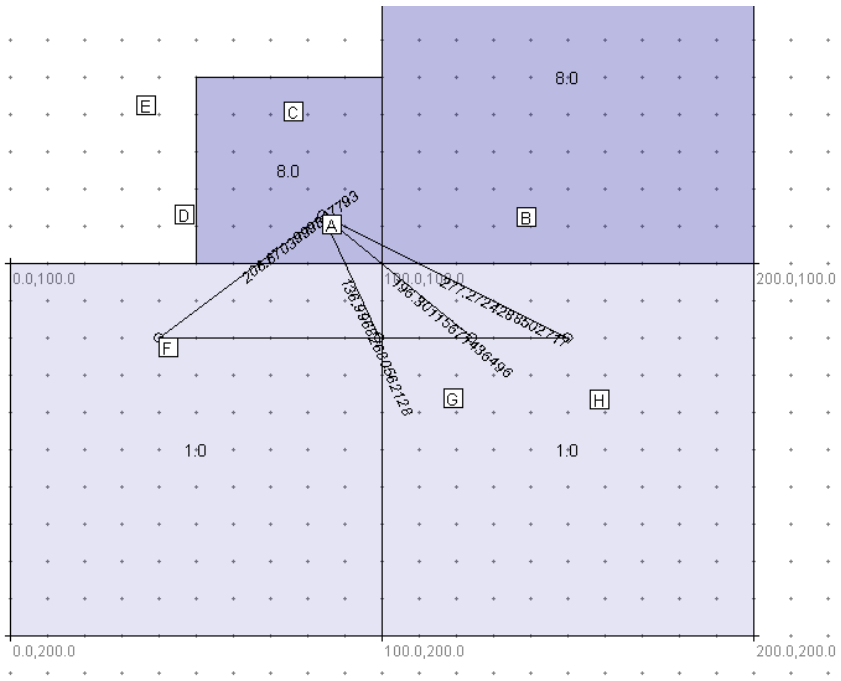

Figure 55: Example Cell Overlap

7.1.4 Network Simulation

All packets are transmitted over a simulated data link layer. This layer transmits packets to all nodes that are within range based on the simulated wireless environment and the maximum ASL setting. The maximum ASL can be any value appropriate to the simulated hardware. For example, this is 85 dBm for standard IEEE 802.11b compatible network adaptors. The simulated wireless environment describes the propagation model of signals over physical space. It represents user-created scenarios of signal propagation over real-world simulated objects. By default, all space has a free-space propagation model. For variations to this, a simulated environment can be created to include signal loss modifiers over an area or boundaries of interference. Over an area, signal loss increases over distance based on a modifier, which is multiplied by the distance a signal travels through that area, reducing the rate of propagation. Free space loss has a modifier of 1.0, whereas a modifier of 2.0 would mean a signal travelling through this simulated environment would travel half the distance as it would in free space. Boundaries of interference simulate barriers of reduced signal propagation, such as walls. A signal passing through a boundary has an immediate loss of signal strength based on the signal loss value of the boundary. The value of boundaries is measured in logical meters. An example simulated wireless environment for testing boundaries with six nodes is shown in Figure 56.

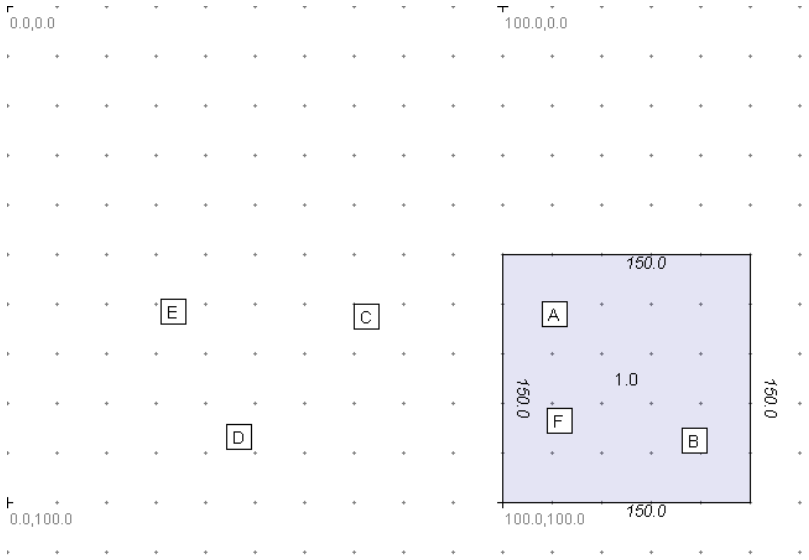

Figure 56: Sample Simulated Wireless Environment

As each packet is successfully transmitted or lost (based on signal propagation), the simulator records what each node's CM predicted the signal loss to be, along with the

actual signal loss based on the simulated wireless map. Figure 57 below shows how a packet would be routed from a transmitting node to nodes within range, based on the simulated wireless environment map for the given scenario.

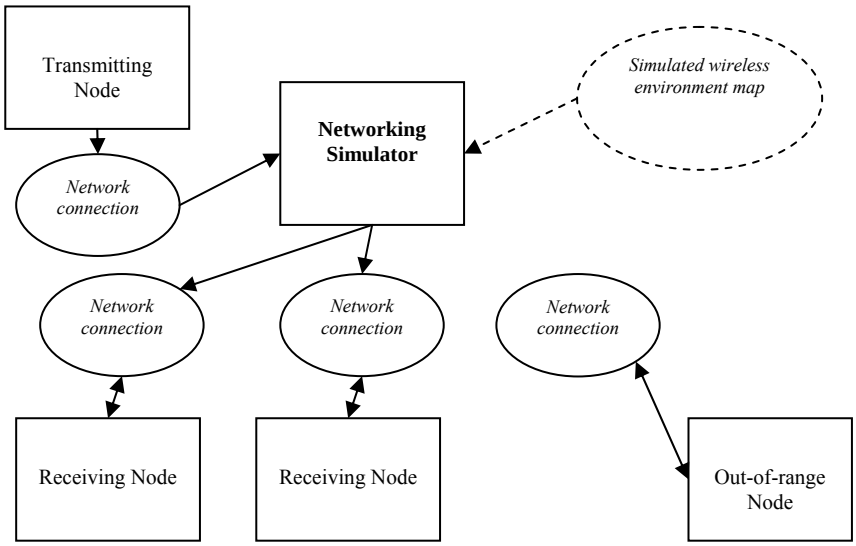

Figure 57: Example Packet Broadcast

For bandwidth simulation, each node maintains a queue of packets which will be transmitted in the next cycle of the simulation (a cycle is typically a small quantity of time, such as 10ms, and is user-configurable). The network simulator ensures all nodes transmit only as many packets as are allowed given the shared medium being used, but does not consider collisions. The total number of packets which may be transmitted is limited based on the bandwidth of all nodes within range as a local shared medium maximum. For example, Figure 58 shows that Node A has three neighbouring nodes. These nodes reside in two clusters, where the cluster represents nodes having connectivity with all other nodes in a cluster. From Node A's viewpoint there are four nodes in its shared space; therefore, it will transmit data to a maximum of $1/4^{th}$ the available bandwidth. Thus, if the bandwidth for this network is 1mb/s, then Node A will transmit at most 250kb/s of data over all cycles in a given second. The maximum bandwidth, B_{max}, per cycle that a node may transmit is calculated as:

$$B_{\max} = \frac{b}{n}\Delta t \qquad (7.1)$$

where b is the total bandwidth (per second) over the wireless medium, n is the number of nodes within range of a node (including that node itself), and Δt is the duration of each cycle (in seconds).

In theory, individual clusters could transmit more in their shared space than the above model suggests, though nodes in multiple clusters would have packet loss where transmissions from two or more clusters were received simultaneously. As the emphasis in this research is not on wireless medium sharing but on predicted routing, the model detailed here has been selected as a practical one. Although this does not take into consideration the multitude of realistic shared medium factors, its only purpose is to ensure that general bandwidth is considered.

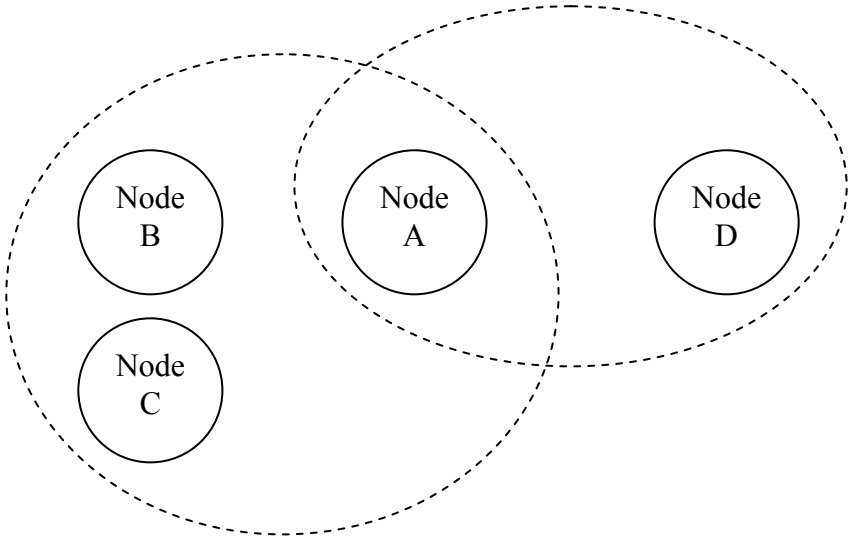

Figure 58: Node Bandwidth Implementation

7.1.5 Routing

Routing is difficult to compare because of the vast number of network topologies which can be created. Each network may also have differing requirements, such as route stability, shortest path routing, or optimised convergence times. For comparison purposes, each routing protocol maintains active connections between every node in the network where a path exists. This suits distance vector-based routing protocols, but not on-demand routing protocols. Modifications have been made to the core routing modules to ensure that, regardless of the routing protocol used, a path to each destination is maintained where available.

Each node in the simulator maintains a routing table. This routing table can be based on any of the routing algorithms described in Chapter 6, or any other wireless ad hoc

routing protocols. For data collection, each routing table saves the connection history for each neighbouring node. The format of entries in the history table is presented in Figure 59. Each entry stores the time at which the route was established, the number of nodes (hops) between the source node and this neighbour, the time this route has been connected, when the route is predicted to terminate, and the path to the destination (if an on-demand routing protocol is being used). Each routing table also maintains a count of *unexpected drop-offs* for every route which was broken before it was predicted to be broken.

route establishment time
hop count
connectivity period (in seconds)
expected route expiry time (in seconds)
path (of nodes)

Figure 59: Connection History Entry Structure

Routing information in the simulator is used primarily for performance analysis and comparison of the CM and FNT algorithms in relation to various routing protocols. The algorithms in this research do not specifically require packets to be transmitted to individual nodes. Rather, they broadcast all information to local neighbouring nodes or to the global network. However, where specific routing protocols require packets to be transmitted to an individual destination node, the network connection for each node will obtain the next hop node on the path to the destination from the implemented routing protocol.

7.1.6 Limits on Node Population

The simulator is designed to run on a single host machine, and has not been designed for clustering or multi-processor systems. Network packet propagation, routing, the FNT and CM algorithms all increase load linearly for each single node, but to a complexity of $O(n^2)$ where all nodes are simulated on a single machine. Due to the computational and memory overhead of the FNT, CM and various routing algorithms, there is a limit on node population if scenarios are to be completed in a reasonable timeframe (up to 24 hours for 8-node populations has been observed). This is further complicated with each scenario being cross-tested with various combinations of settings (31 setting variations were used in total over each of the scenarios presented in the next section). Therefore,

testing of medium and large-scale simulations has not been possible for FNT and CM algorithms, limiting the extent to which they can be adequately analysed.

7.2 Summary

The simulator was developed for this thesis to provide a means of testing wireless network protocols based on signal loss propagation characteristics. The component-based architecture of the simulator allows testing to be developed and performed on a wide variety of algorithms. Specifically, protocols related to signal loss mapping, autonomous nodes and prediction can be studied. WANS is able to simulate node protocols and movement, signal propagation and network connectivity on which wireless protocols can be based. A number of statistical components are provided to collect information on signal loss maps, prediction, and routing protocols. A number of operational snapshots are also provided as a means of observing and analysing protocols in action.

It is with this information that the 14 primary scenarios (based on 8 scenario environments with varying node populations) over 31 variations of algorithm settings are analysed in the upcoming results chapter.

Chapter 8: Simulation Results

The aim of this chapter is to test if prediction is beneficial to routing algorithms, and if both CM and FNT concepts produce more accurate predictions than previous work.

This chapter is divided as follows. Firstly, the scenarios are presented, designed to test the various aspects of the algorithms in this thesis. Then each of the core components, CM, FNT, and implementation in routing protocols are described and tested in separate sections. The overall results are then summarised.

8.1 Scenarios

Simulations of wireless ad hoc routing protocols often use randomisation algorithms to place nodes within an area (such as in [HJ01] and [LK00]). However, these examples use a free-space signal propagation model, and no wireless environment is simulated. The scenarios designed for this thesis are all manually designed because of the need for simulated wireless environments. Signal propagation is affected by the wireless environment in which nodes operate. Such scenarios, while random to an extent in real life, can be better designed and have previously been designed manually ([HSS03], [YAS03]). Node placement and movement in such scenarios can remain random to an extent, although to test specific map coverage there is also a requirement for specific node placement. The scenarios presented in this thesis are entirely manually designed, including node placement and movement, and represent typical situations which occur in realistic settings.

A total of eight scenarios have been designed to both represent realistic environments (through wireless signal mapping experiments) and test the various aspects of the algorithms presented in this thesis. Two of these scenarios are specifically designed to test the boundaries concept of CMs. Some scenarios have been tested with a varying number of nodes to further analyse scenarios while gaining an insight into the effects of network population. Each of the scenario descriptions below includes a scenario diagram, which details node placement, node movement, simulated network environment and grid

coordinates. Node movement is represented by task lines. Times for the beginning and ending points of each task line are not included, to reduce clutter. The simulated network environment is represented using shaded areas of reduced signal propagation, with the signal loss modifier and boundary logical distance values shown. All diagrams are to scale, with grid markers at 10m intervals.

These scenarios have been designed to represent as many individual circumstances that occur in wireless ad hoc networks while maintaining general simplicity. Few nodes were used due to the hardware requirements of simulating many nodes on a single server. This is due to the amount of logging and data analysis that each simulation creates, so that data can be reviewed after experiments.

8.1.1 Scenario 1

This scenario's primary purpose is to test to what extent a simulated building can be mapped (Figure 60). A building is placed in the centre of the map, with up to four nodes surrounding it, and up to two nodes inside it. The surrounding nodes travel around or through the building, while the nodes inside the building remain stationary. This scenario was executed using 4, 5 and 6 node populations, as described in Table 8.1.

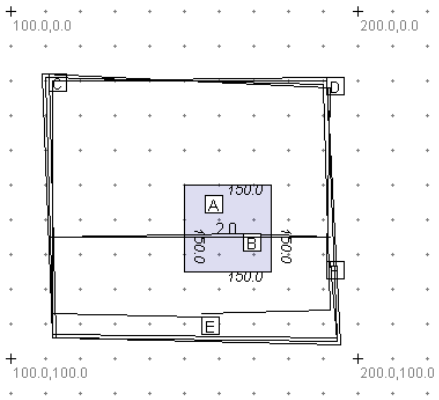

Figure 60: Simulated Wireless Map for Scenario 1

Table 8.1: Node Population

4 nodes	B, C, E, F
5 nodes	B, C, D, E, F
6 nodes	A, B, C, D, E, F

8.1.2 Scenario 2

Scenario 2 extends Scenario 1 to experiment with the accuracy of the algorithms in this thesis where two simulated buildings exist (see Figure 61). Two buildings are placed on the map approximately 40 meters apart. Two nodes, E and F, remain stationary in the

smaller building. Up to six other nodes travel in varying patterns around and between the two buildings. This scenario was executed using 6 and 8 node populations, as described in Table 8.2.

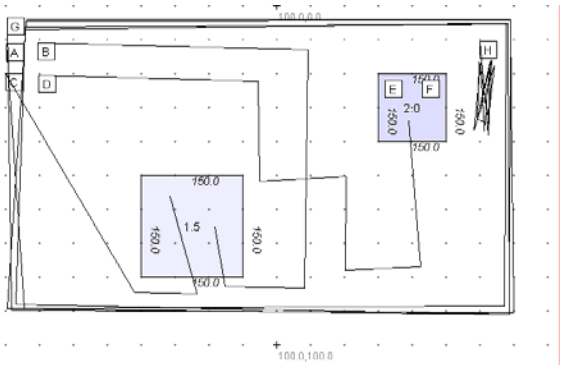

Figure 61: Simulated Wireless Map for Scenario 2

Table 8.2: Node Population

6 nodes	A, B, C, D, E, F
8 nodes	A, B, C, D, E, F, G, H

8.1.3 Scenario 3

Scenario 3 was designed to validate the boundary algorithms of the CM (Figure 62). A single building is simulated with four walls of 150 logical meter loss. Inside the building there is no reduced signal propagation (emulating a large open shed). No nodes are moving in this experiment so as to specifically test certain boundary properties. Scenario 4 adds moving nodes to the same layout as this scenario. This scenario was executed using 3, 4 and 6 node populations, as described in Table 8.3.

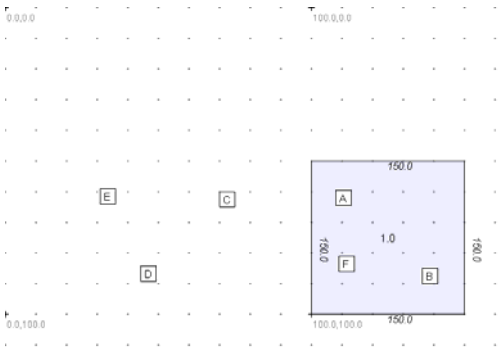

Figure 62: Simulated Wireless Map for Scenario 3

Table 8.3: Node Population

3 nodes	A, B, C
4 nodes	A, B, C, D
6 nodes	A, B, C, D, E, F

8.1.4 Scenario 4

Scenario 4 expands on Scenario 3 by introducing two moving nodes and two more stationary nodes. These nodes circumnavigate the building once (Figure 63). This scenario was executed using 6 and 8 node populations, as described in Table 8.4.

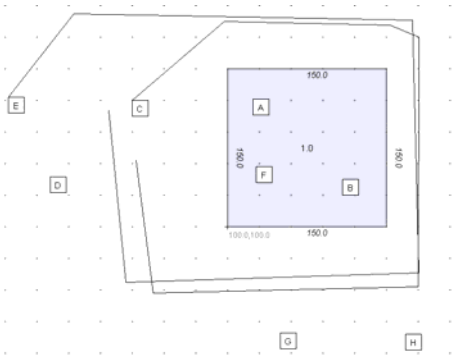

Figure 63: Simulated Wireless Map for Scenario 4

Table 8.4: Node Population

6 nodes	A, B, C, D, E, F
8 nodes	A, B, C, D, E, F, G, H

8.1.5 Scenario 5

Scenario 5 was designed to determine the effectiveness of the FNT algorithms (Figure 64). Eight nodes are placed within or outside a simulated area with a very high signal loss modifier of 8.0. This high signal loss ensures connectivity between nodes is broken through the building. Three of the nodes (C, E, and F) are moving around the building for the purpose of generating a FNT with varying levels of connectivity prediction.

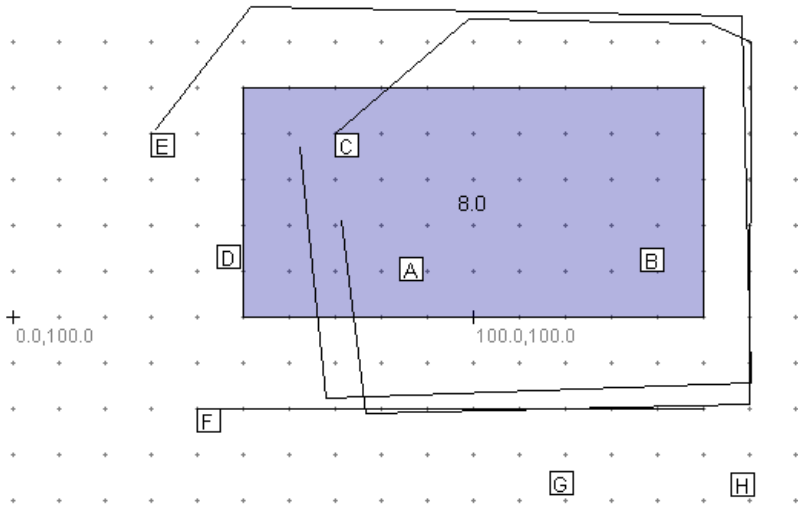

Figure 64: Simulated Wireless Map for Scenario 5

8.1.6 Scenario 6

Scenario 6 contains only four nodes with two simulated wireless propagation areas (Figure 65). To the left is an area with a small signal loss modifier, based on signal loss measurements found in a local sparse forest. On the right-hand side is a building with thick walls (having a logical distance of 400 meters) and a high signal loss modifier of 5.0. All four nodes are in motion in this scenario, with nodes C and D moving from outside the building over to the forest area, and nodes A and B moving within the building.

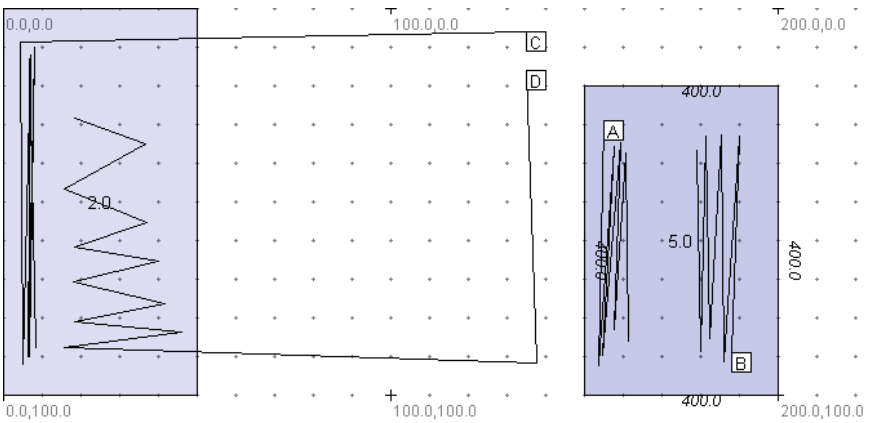

Figure 65: Simulated Wireless Map for Scenario 6

8.1.7 Scenario 7

Scenario 7 is based on a real neighbourhood street (the scenario diagram is included in Figure 66). It is based on signal loss readings measured using 802.11b equipment. Six nodes are based around a T-intersection of the street, with 11 houses. Each house has walls having a logical distance of 50 meters, and an interior signal loss modifier of 2.0. Three of the nodes, A, B, and F, travel along the streets.

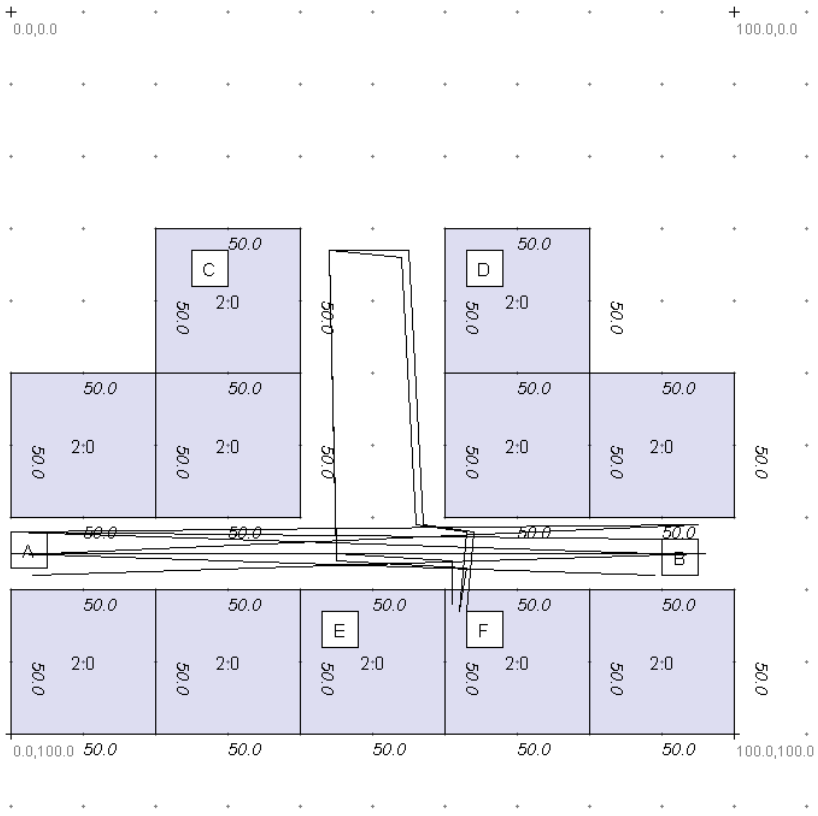

Figure 66: Simulated Wireless Map for Scenario 7

8.1.8 Scenario 8

Scenario 8 is included as a free-space signal loss scenario (Figure 67). It contains 6 nodes, 3 of which have task paths. Due to the distances between nodes (up to 300 meters), connectivity is naturally broken, however there exists no reduced signal propagation. Consequentially, a CM map should not develop or represent anything other than free-space loss. This scenario's purpose is to analyse the performance of FNT algorithms without the need for a CM.

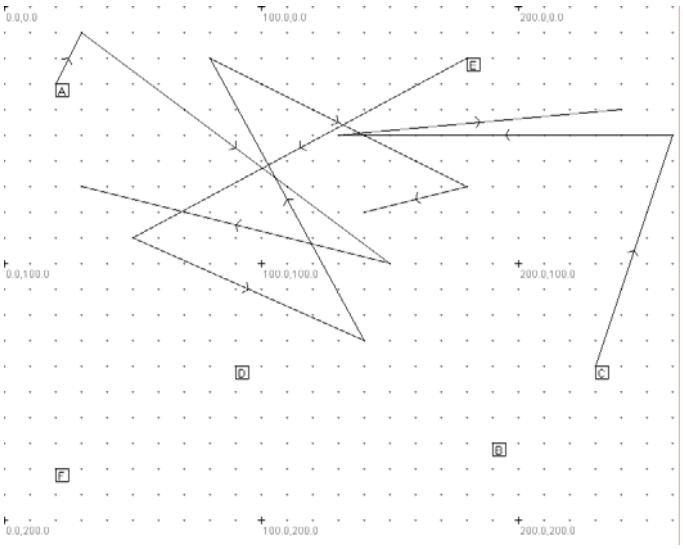

Figure 67: Simulated Wireless Map for Scenario 8

8.2 Communication Map

Several scenarios have been created to validate the concept of the CM, while also identifying its weaknesses. The field of wireless communication has an unlimited number of practical scenarios. This presents a significant challenge in representing a broad spectrum of possible scenarios to gauge the overall effectiveness of the CM. However, scenarios have been developed which focus on specific situations that the CM should be capable of handling in order to validate the accuracy of the CM.

Generating a constantly-accurate CM is not feasible without a vast grid array of fixed-position sensors. The CM is only able to map signal loss over areas where signals have been successfully transmitted. This is sufficient where nodes frequent common areas, as maps are shared among all nodes. Therefore the generated CMs may produce accurate estimations without necessarily being similar to the actual signal propagation environment maps as nodes may not travel over the entire map. The accuracy of the CM compared with the signal propagation map is thus not studied. Instead, the accuracy of each predicted signal in relation to each actual signal received at the same time is studied. In this way, the CM's performance is measured based on its ability to predict the same signal loss as will actually be received. The results for CM testing use measurements of error between estimated and actual signal loss over time as the primary means to test CM performance. The CM is constructed using signal loss readings from all nodes in the network due to map sharing, which is tested first.

8.2.1 Signal Loss Mapping Techniques

Algorithm 4.3 included a signal loss mapping technique attributing increased signal loss in relation to the quantity of signal loss each cell has recorded. This weighting of signal loss to areas which are already predicting high loss is designed to speed the development of the CM. Experiments comparing Algorithm 4.3's signal loss mapping technique with a generic distance-weighted technique are necessary to validate Algorithm 4.3. CM accuracy over time is presented for Scenarios 5 and 7 in Figure 68 and Figure 69. Both graph the accuracy of CM predictions using two versions of Algorithm 4.3. The weighted version operates to the specification of Algorithm 4.3, using the technique of attributing signal loss to cells with high signal loss modifiers. The generic version divides signal loss over multiple cells based entirely on the distance that a signal travels through a cell.

Results show for both scenarios that using a weighted technique reduces the average error considerably. Experiments across all scenarios 1 to 8 also resulted in the weighted technique producing less average error than the generic technique. The weighted approach allows signal loss modifiers to be derived for cells that are never entered by nodes, but may reside between two nodes' line of sight. Cell accuracy is therefore greater with the weighted approach.

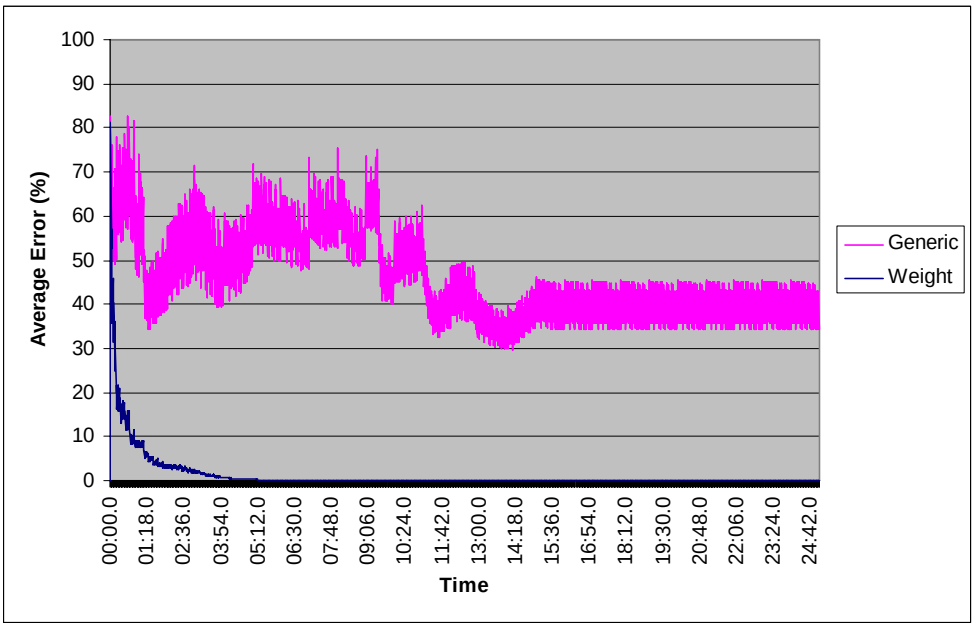

Figure 68: Average CM Error for Scenario 5

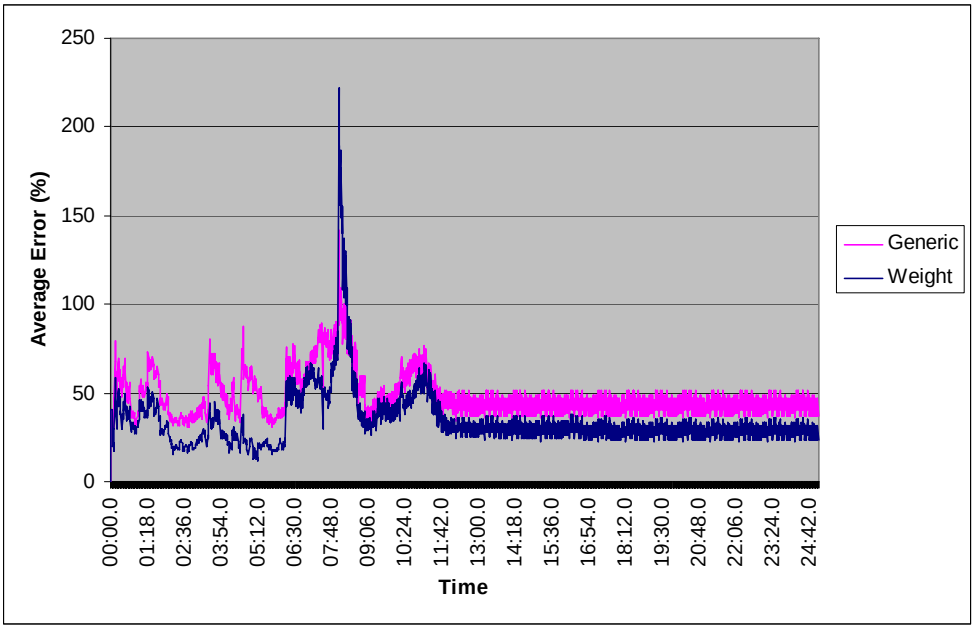

Figure 69: Average CM Error for Scenario 7

8.2.2 Adaptability to Changing Environments

The CM has been designed to adapt to changes in the signal propagation environment over time. In this experiment, Scenario 6 has been adapted such that the signal propagation environment changes over time. The first 5 minutes of the simulation are identical to that of Scenario 6, except that signal loss on the right-hand object's boundaries has been removed for simplicity (Figure 71 on the following page). After 5 minutes, the left-hand object's signal loss modifier increases, while the right-hand object's signal loss modifier decreases (Figure 73). After 10 minutes, the left-hand object returns to original signal loss and moves to the right. The right-hand object's signal loss modifier also returns to original signal loss (Figure 75). Accuracy results over time compared with the original Scenario 6 are shown in Figure 70. The resulting CMs from Node A's perspective at 5, 10 and 15 minutes are shown in Figure 72, Figure 74 and Figure 76. Figure 70 graphs the average error over time between the original Scenario 6 with a stationary map and the modified Scenario 6 with a map of moving signal propagation characteristics. As can be seen from the resulting CMs and Figure 70, the CM is able to adapt to the changes within the 5 minute period before the next change.

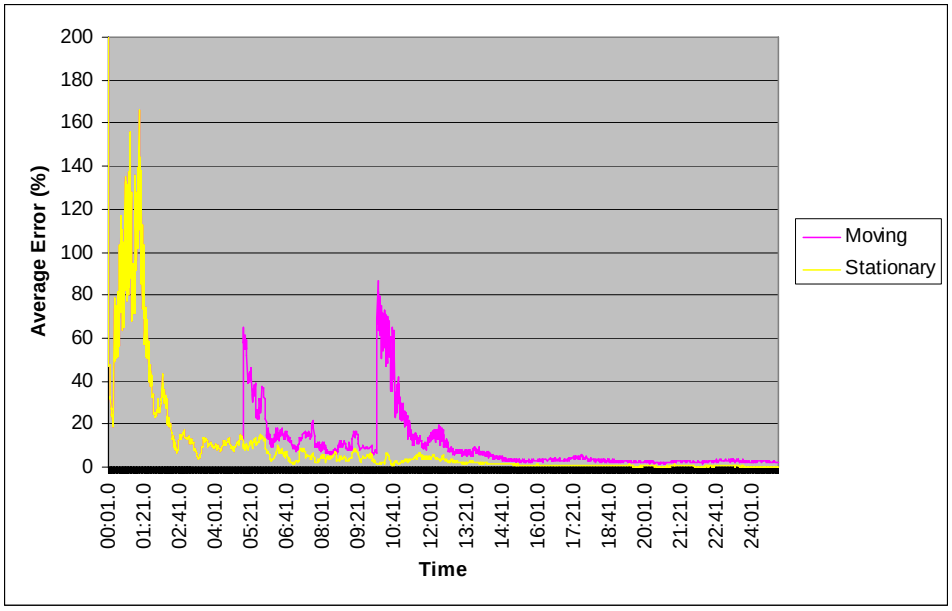

Figure 70: Average CM Error for Scenario 6

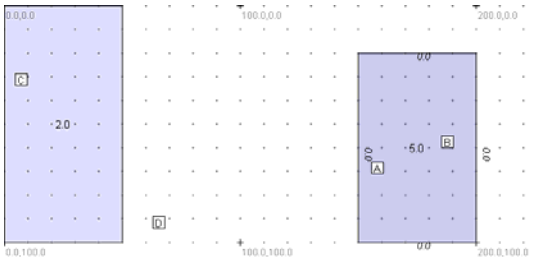

Figure 71: SWM at 5 minutes

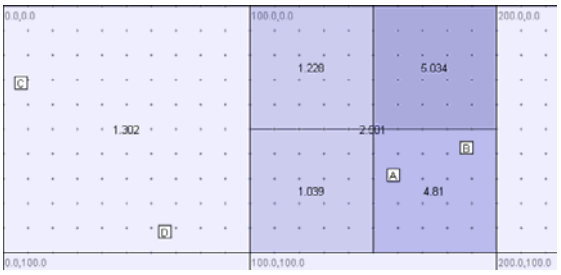

Figure 72: Node A's CM at 5 minutes

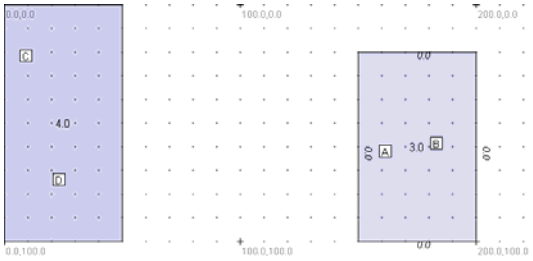

Figure 73: SWM at 10 minutes

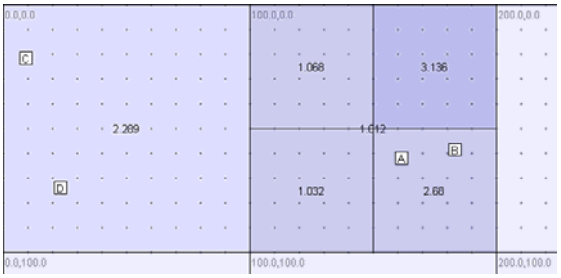

Figure 74: Node A's CM at 10 minutes

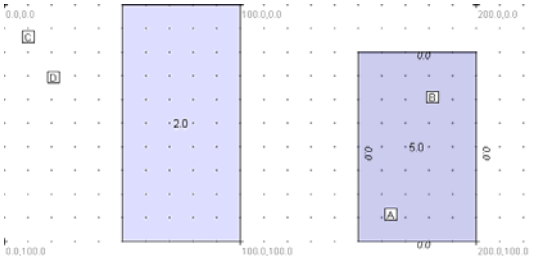

Figure 75: SWM at 15 minutes

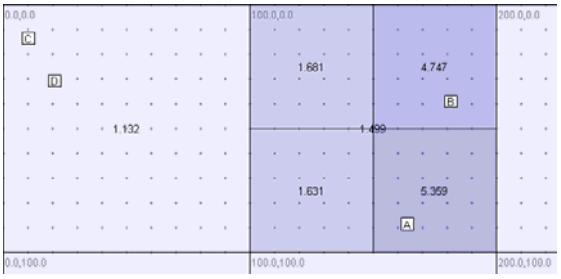

Figure 76: Node A's CM at 15 minutes

8.2.3 Communication Map Sharing

The ability for each CM to develop over time relies on signal loss readings and positional information from all nodes in the network. In doing so, nodes with no movement at all are still able to create an accurate CM, which can in turn be used should such nodes later move and require prediction. It is important that each node maintain a CM similar to all other nodes in the network. Without doing so, there is potential for predictions of one node to be different from predictions of another node. More importantly, if several CMs are different, then the goal of an "accurate" CM cannot justifiably be reached, as nodes would not form a shared view of the signal propagation topology. Figure 77 shows the average differences between all CMs for each scenario averaged over the duration of each simulation. As Figure 77 shows, there is typically less than 0.2% difference between individual CMs for any scenario tested. Scenario 8, requiring only a free-space loss CM, has no differences at all.

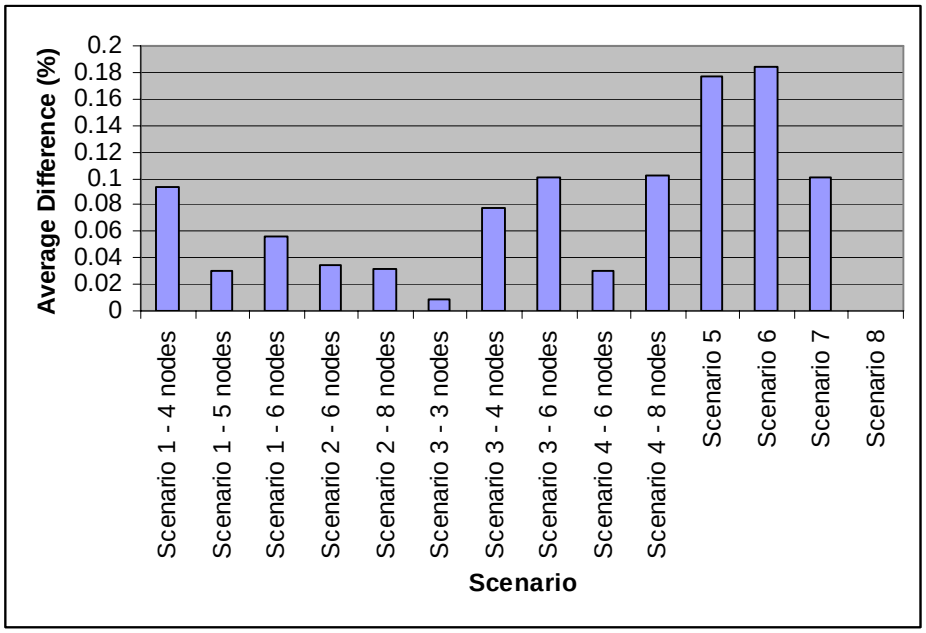

Figure 77: CM Differences for all Scenarios

8.2.4 Boundaries and DCS

The choice of implementation between boundaries and the basic CM is significant. The use of boundaries is a modification to the original algorithm as an attempt to improve on the CM solution. The proposed solution uses boundaries as a signal loss modifier between areas where the signal loss does not change over the distance the signal travels,

such as a building wall. This section studies the results of how implementing boundaries affects the performance accuracy of the CM.

In the majority of these experiments, two sets of results are generated for each scenario. One set is generated using the basic algorithm without boundaries, and one set is generated using the algorithm implementing boundaries. In each set, three Default Cell Size (DCS) settings are implemented: 25 meters, 50 meters, and 100 meters (the default). This is done to determine how reducing the DCS affects accuracy. In theory, a smaller DCS should improve CM accuracy to that of a CM implementing boundaries. This is due to the fact that a smaller DCS enables the CM algorithms to more accurately map signal loss immediately. With larger DCS settings, the loss accounted for in artificial boundaries is averaged into the larger cells.

Scenario 4 was created specifically to test the boundary concept. The scenario consists of a single building, with perfect free-space loss within the building (for example in an empty warehouse) and with walls of 150 logical meter loss (as estimated from signal loss measurements taken of a thin wooden wall for this thesis). Three nodes are placed within the building, with a further five nodes outside it. Two of the nodes, E and C, circumnavigate the building. Figure 78 below illustrates this.

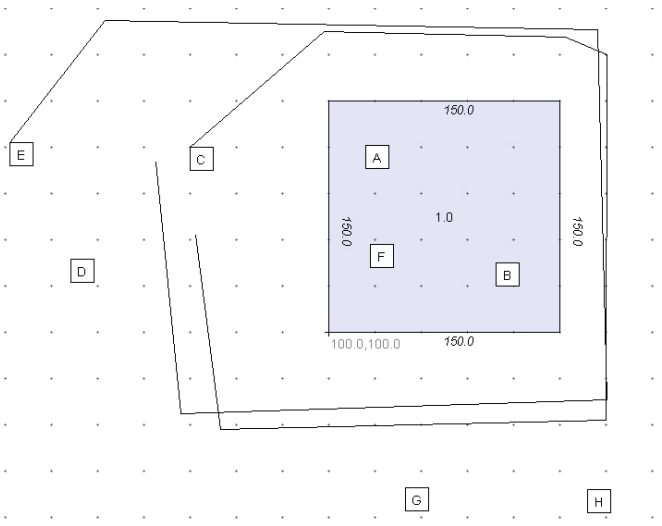

Figure 78: Simulated Wireless Map for Scenario 4

The results from these experiments are shown in Figure 79 and Figure 80. Without implementing boundaries, the average error stabilises to between 35% and 50%, when using a DCS of 100 metres (Figure 79). Reducing the DCS by half improves accuracy

considerably, with further reduction possible if the DCS is lowered to 25 metres. This demonstrates that a reduced DCS allows the CM to map the effects of simulated boundaries similar to the way the boundaries algorithm does (when compared with Figure 80). Implementing boundaries shows significantly improved accuracy.

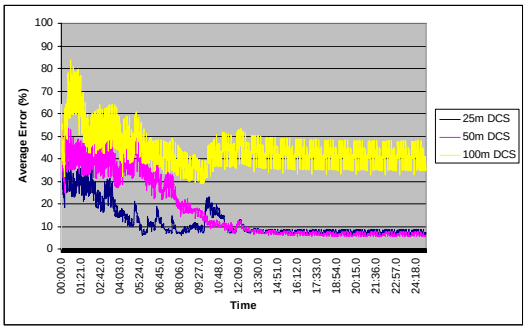

Figure 79: Scenario 4 without Boundaries

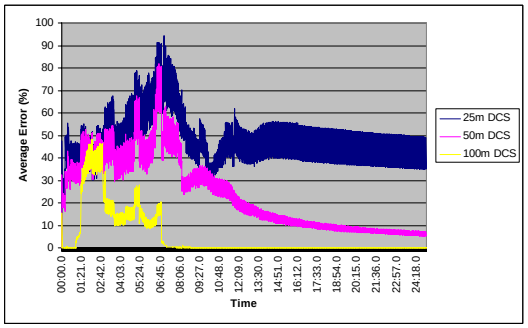

Figure 80: Scenario 4 with Boundaries

However, reducing the DCS in Scenario 4 with boundaries does not improve accuracy, but has the reverse effect. Figure 81 through to Figure 84 illustrate the CMs at the end of each scenario from Node A's viewpoint. Figure 81 shows the CM where a DCS of 25m and no boundaries were used. The CM algorithm represented the signal loss surrounding nodes A, B, and F by using the smaller DCS as a boundary representation itself. However, even with 8 nodes the CM does not perfectly represent the shape of the simulated building, nor gives accurate signal loss.

Figure 82 shows the typical 100m DCS with boundaries, with Figure 83 and Figure 84 reducing the DCS to 50m and 25m respectively. The smaller DCS with boundaries actually hinders the boundary development, as boundaries are more difficult to develop than cells. With the smaller DCS in the boundary examples, there are too many objects where signals can be mapped. As signals being mapped are averaged over all objects along the assumed signal's path, the greater number of objects requires a greater quantity of node movement to further correlate signal loss more accurately. This is infeasible using only eight nodes.

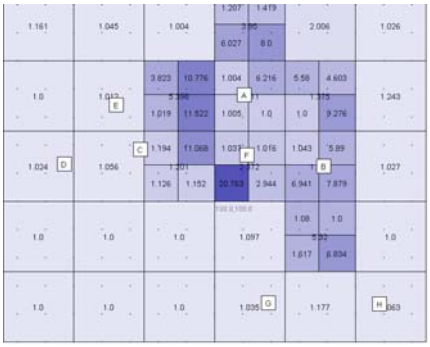

Figure 81: Scenario 4: 25m DCS without Boundaries

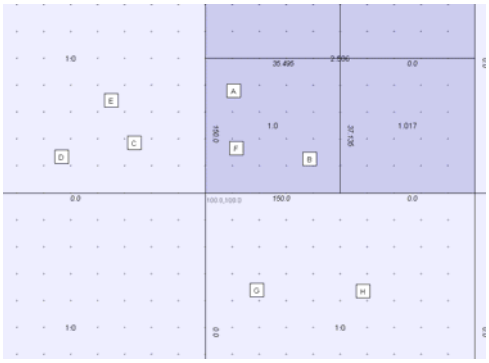

Figure 82: Scenario 4: 100m DCS with Boundaries

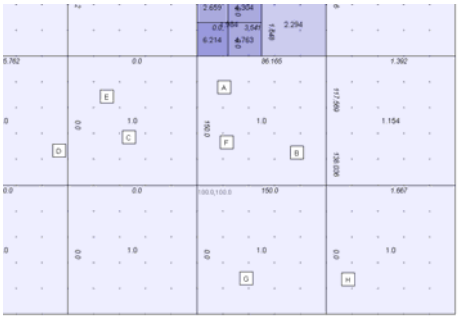

Figure 83: Scenario 4: 50m DCS with Boundaries

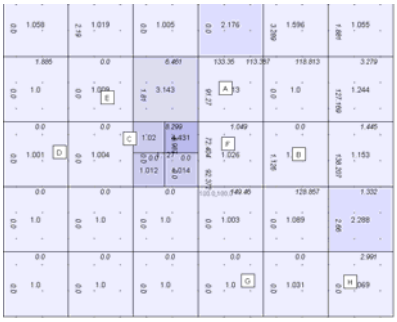

Figure 84: Scenario 4: 25m DCS with Boundaries

Scenario 4 was designed to study a single object requiring boundaries. Scenarios 5 and 6 were also designed to study boundaries, but are slightly more detailed. The same experiments from Scenario 4 were run on Scenarios 5 and 6 to further study the effectiveness of implementing boundaries. Results are shown in Figure 85 and Figure 86. Surprisingly, these experiments indicate that overall the varying DCS settings and the implementation of boundaries make little difference to the overall accuracy.

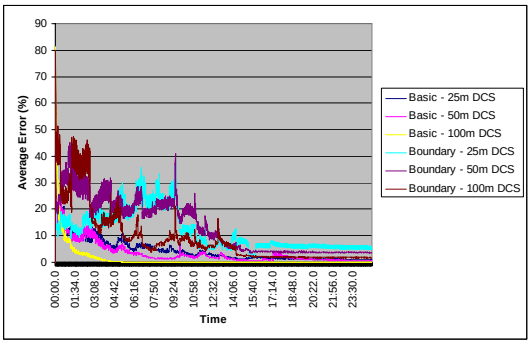

Figure 85: CM Accuracy of Scenario 5

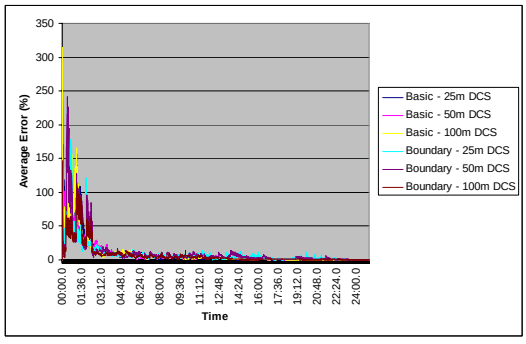

Figure 86: CM Accuracy of Scenario 6

The experiments were then run on Scenario 7, which is based on a real street. As with Scenarios 5 and 6, all settings produced similar results (Figure 87), indicating that in more

realistic, complex environments implementing boundaries has little effect. The CMs at the end of the simulations of a 25m DCS without boundaries and a 100m DCS with boundaries experiment are shown in Figure 88 and Figure 89. The favourable results in these experiments can be attributed to the low amount of node movement, where only three nodes are moving in relatively fixed movement patterns (refer to Figure 66 earlier). This goes to illustrate that a CM does not have to be physically similar to the simulated wireless environment map in order to produce favourable results. Only a CM which represents the signal loss as it has been used and will be used in the future is required for accuracy.

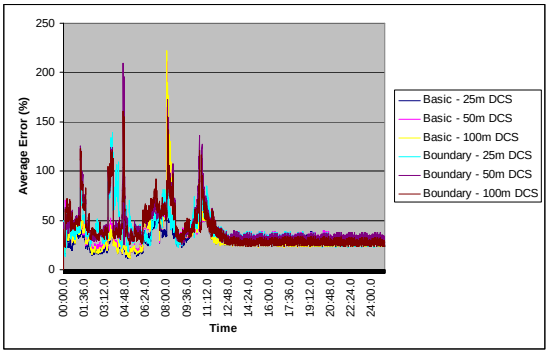

Figure 87: CM Accuracy of Scenario 7

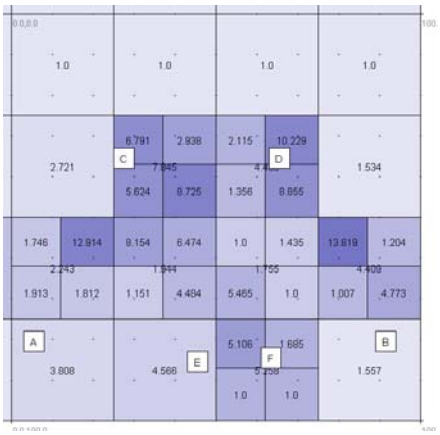

Figure 88: 25m DCS CM without Boundaries for Scenario 7

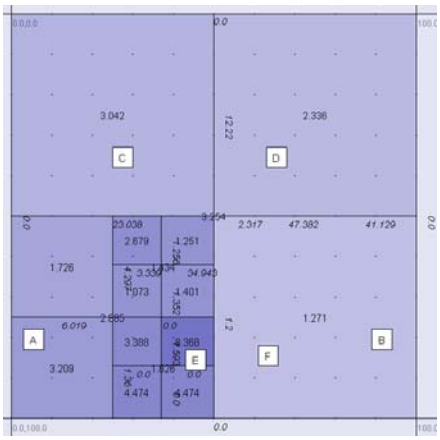

Figure 89: 100m DCS CM with Boundaries for Scenario 7

To gauge the overall effectiveness of boundaries and of DCS values, all scenarios were run with all settings, and overall average results were graphed in groups of settings, shown in Figure 90 below. Results are ordered left-to-right in order of average accuracy for each setting. Using a 25m DCS without implementing boundaries produces marginally better results overall. However, this must be understood in context. Some

scenarios have been designed to test specific algorithms, and as such will result in some experiments producing extremely favourable results. Overall the majority of DCS and boundary settings do produce reasonably consistent results among all experiments. The use of boundaries does not offer a significant accuracy improvement overall compared to an implementation without boundaries and a lower DCS. Given the overhead of boundary calculation and the increased bandwidth (presented in detail later) of boundary information in CM broadcasts, the implementation of boundaries is therefore not recommended. The DCS should be selected based on the target environment that nodes will operate in, and varies between situations. Most cluttered environments (such as urban environments) will benefit from a reduced DCS, as this allows smaller areas of similar signal propagation characteristics to be mapped faster.

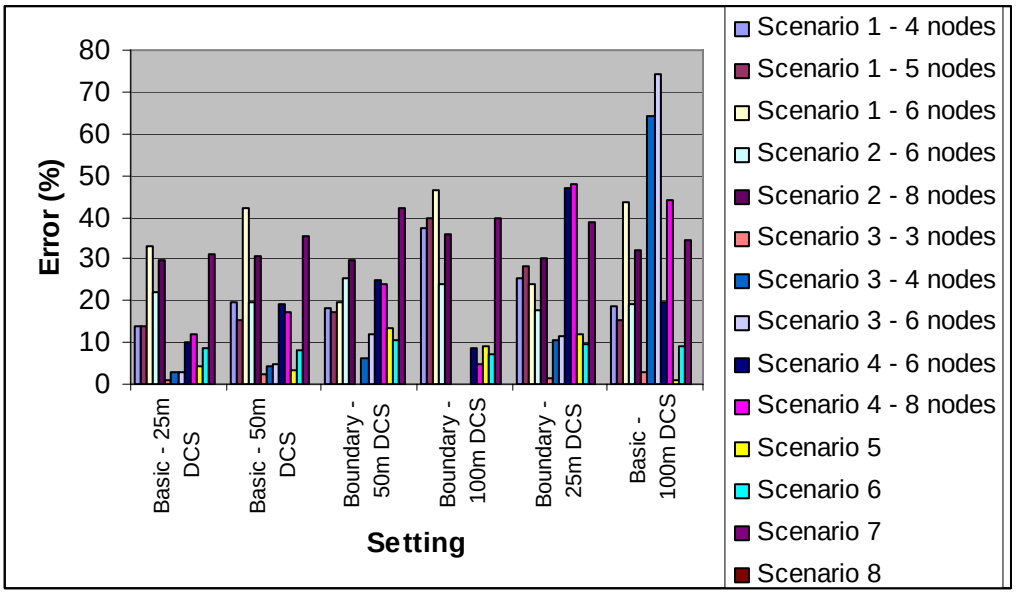

Figure 90: CM Accuracy for all Scenarios

8.2.5 Number of Nodes

Scenarios 1 to 4 were tested to determine if the number of nodes has an effect on the accuracy of the CM. These are the four main scenarios where the basic layout would allow the number of nodes to play an influential role on CM development, but without new areas being discovered. In more complex scenarios (such as Scenario 7), adding more nodes often only leads to new areas being discovered where loss must be attributed. Consequently, a small group of nodes can only cover a small area. A much larger group of nodes can cover a larger area, though the hardware requirements for testing such

scenarios were impractical for this research. Scenarios 3 and 4 focus more on boundary testing, but are included as further simple examples. All four scenarios are using a DCS of 100m, without implementing boundaries.

The results in all experiments were different than anticipated. In theory, increasing the number of nodes should increase the accuracy of the CM. However, in almost all scenarios tested, an increase in the number of nodes had an adverse effect on accuracy. This is because an increase in nodes in these scenarios increases the area over which signal loss must be predicted. By increasing the area there is a greater potential for inaccurate signal loss estimations, decreasing overall accuracy. Scenario 1 was first tested, with results shown in Figure 91. In Figure 91, using both 4 and 5 nodes resulted in very similar results, with 5 nodes performing slightly better, as would be expected. Using 6 nodes, however, almost doubled the overall average error. To verify that using a DCS of 100m was not the problem, Scenario 1 was tested again using a DCS of 25m, (Figure 92), with the same trend emerging.

The resulting CMs at the end of each simulation are presented. Figure 93, Figure 94, and Figure 95 show each of the resulting CMs of Node B as it perceived the simulated environment to be (shown in Figure 96). Node B was used as Node A does not exist in all node population variations. In the case of 6 nodes, the addition of Node A inside the simulated building causes excess loss to be attributed between Node A and any other node. Any cells between these two points are consequentially allocated a portion of the signal loss, which cannot be accurately mapped due to the signal loss mostly coming from the walls of the simulated building. As other nodes do not lie between the areas without this loss, the areas outside the building are not corrected, and the signal loss is spread outside the building. Due to the lack of nodes with the same cells outside the building, inaccurate signal loss averages were never corrected. This is a difficulty in creating the CM, where location coverage is an important but difficult requirement.

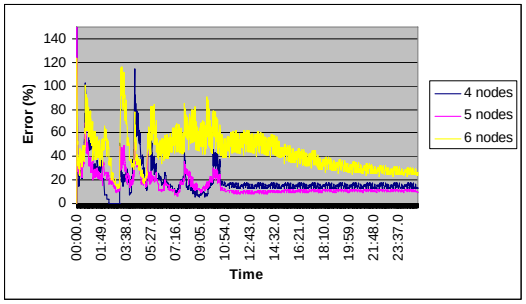

Figure 91: CM Accuracy for Scenario 1 using a 100m DCS

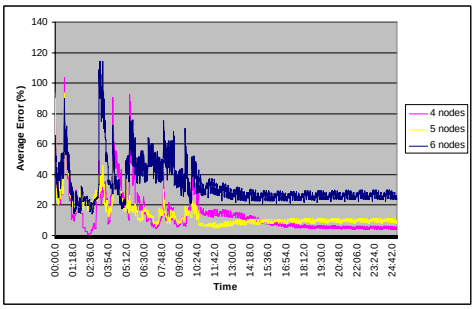

Figure 92: CM Accuracy for Scenario 1 using a 25m DCS

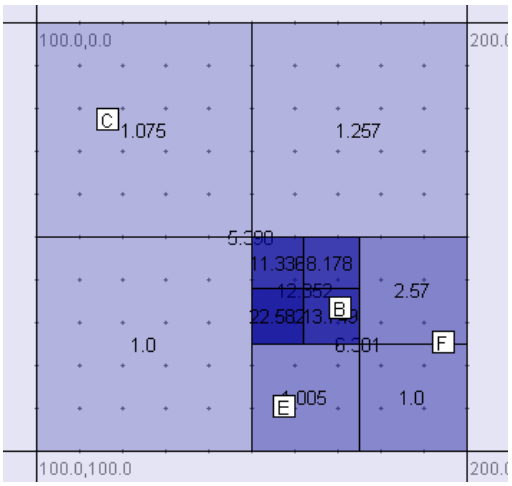

Figure 93: Sample CM of Scenario 1 - 4 nodes

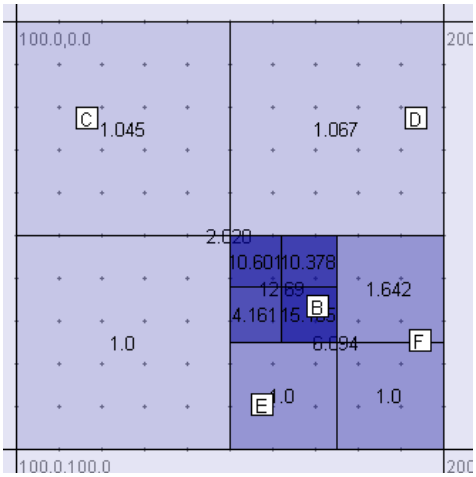

Figure 94: Sample CM of Scenario 1 - 5 nodes

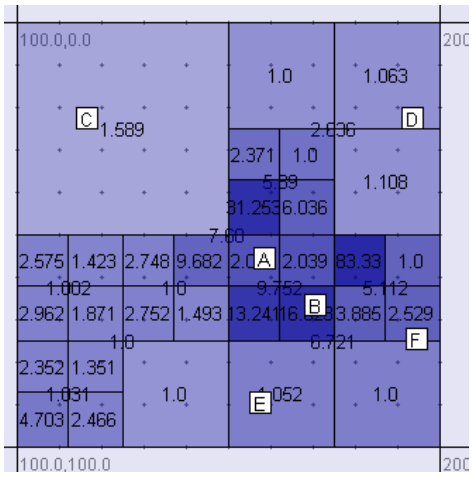

Figure 95: Sample CM of Scenario 1 - 6 nodes

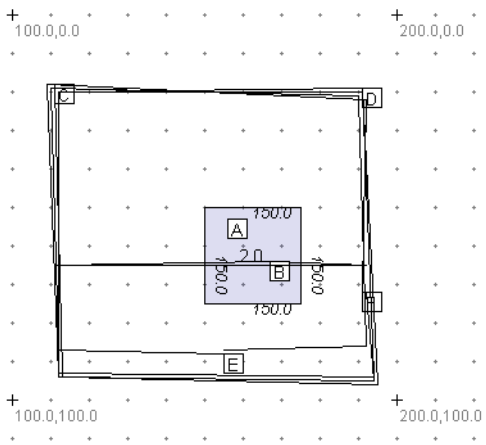

Figure 96: Simulated Wireless Map for Scenario 1

Results from Scenarios 2, 3, and 4 (Figure 97, Figure 98, and Figure 99) show the same pattern, with an increase in node population also increasing CM error. In all scenarios, the loss is attributed to inadequate node coverage to correct inaccuracies in signal loss. This inaccurate signal loss mapping arises due to boundaries of signal loss existing in the wireless propagation environment. In Scenarios 1 to 4, each scenario contained buildings

with walls of signal loss. When this is combined with nodes within the building, these nodes contribute much of the loss within the building, and often loss directly outside the building (as evidenced in the CM of Scenario 1 using 6 nodes, in Figure 95). In effect, the CM is simulating these boundaries of signal loss using small cells directly within and as far outside the building as possible. Where nodes exist outside the building to correct signal loss mappings, such as between nodes C and D, between nodes D and F, and between nodes E and F, the signal loss mappings are corrected. But where there is not enough node coverage, such as immediately left of nodes A and B, signal loss mappings are not corrected. Furthermore, signal loss within the building is not accurately adjusted, as cells are unable to subdivide further. Allowing cells to subdivide to too fine a resolution greatly increases bandwidth and reduces the ability of the CM to average areas as a whole.

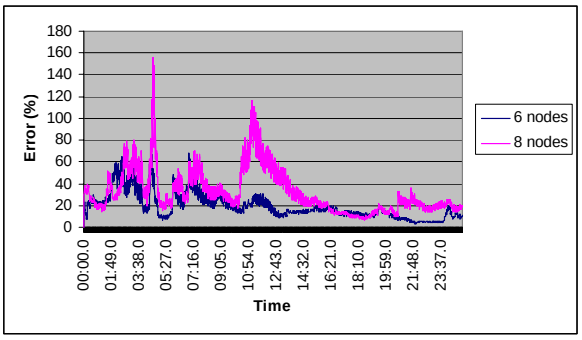

Figure 97: CM Accuracy for Scenario 2

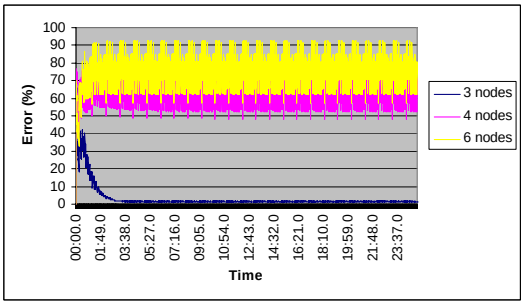

Figure 98: CM Accuracy for Scenario 3

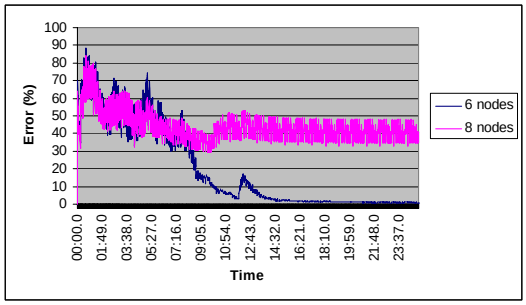

Figure 99: CM Accuracy for Scenario 4

8.2.6 Time Blocks

The Time Block design was introduced in Chapter 4 as a method of averaging signal losses for each cell. It uses two parameters, the duration of each Time Block (in seconds), and the quantity of Time Blocks averaged and shared with neighbouring nodes in total. Shorter Time Blocks will allow Time Blocks to be propagated through the network faster,

which is beneficial in rapidly-changing environments or where node movement encounters new areas often. The quantity of Time Blocks determines to what level older information is considered relevant, which is helpful if there are short anomalies in wireless transmissions, but also prevents more recent information from being immediately accepted.

Figure 100 below averages the results of 14 experiments using 7 different Time Block settings. The Time Block settings are shown left-to-right in order of the average of results for all experiments for each setting, lowest to highest (where lowest is better). The graph shows the average error over each scenario's entire duration given the Time Block settings used.

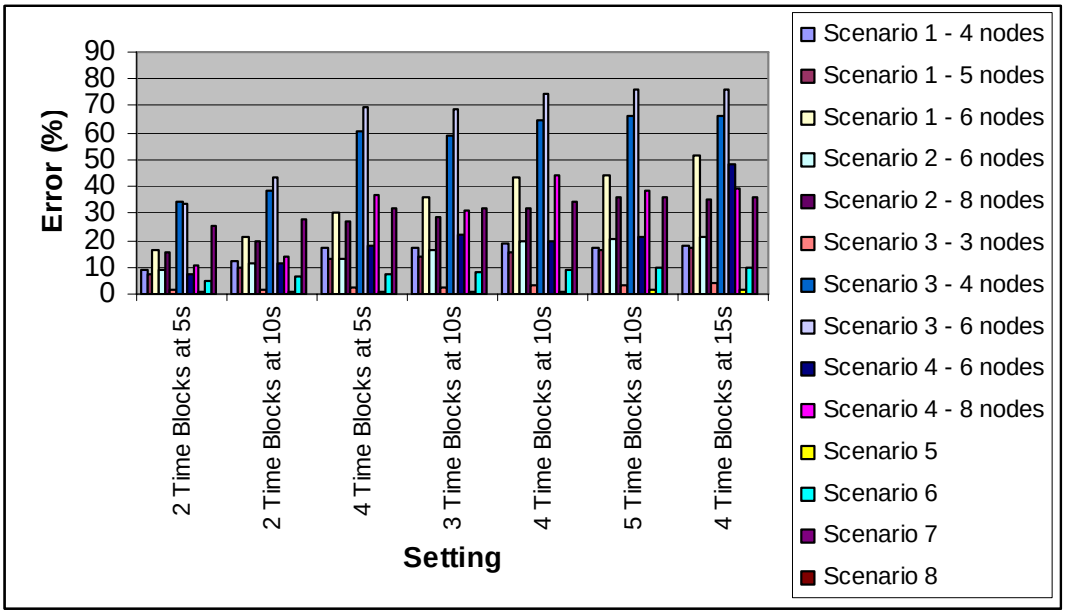

Figure 100: Performance of Time Settings on CM Accuracy

The results show that performance improves by up to 100% in some scenarios with a reduced quantity of Time Blocks and Time Block duration. Using settings of 2 × 5 second Time Blocks produces the best results, followed by 2 × 10 second Time Blocks, and then 4 × 5 second Time Blocks. Whether the Time Block length is reduced or the number of Time Blocks is reduced leads to an improvement in accuracy of the Communication Map. The average error for Scenario 3 both with 4 nodes and 6 nodes is significantly higher than all other scenarios. Scenario 3 has high usage of boundaries which when using a CM without boundaries makes it difficult to accurately map signal loss.

8.2.7 MCSD and MCMD

The Minimum Cell Subdivide Difference (MCSD) and Minimum Cell Merge Difference (MCMD) values have also been considered. The MCSD defines the smallest difference between the highest and lowest signal loss modifier of potential cells required for an actual cell to subdivide. The MCMD defines the difference between highest and lowest signal loss modifiers of four actual cells required for a parent cell to merge the actual cells back together. By default, all scenarios use a MCSD of 3.0 and a MCMD of 2.0. These have been modified in two groups of settings; the first uses a MCSD of 2.0 and a MCMD of 1.0, and the second uses a MCSD of 1.0 and a MCMD of 0.0. These are all compared with the default settings of a MCSD of 3.0 and a MCMD of 2.0 in Figure 101 below.

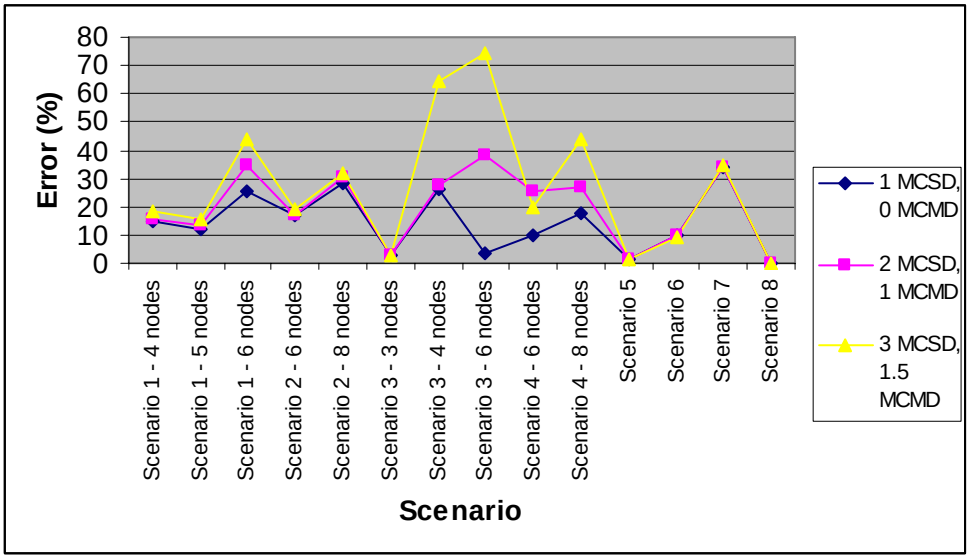

Figure 101: Accuracy of MCSD and MCMD Settings

The results from these experiments show that lowering the MCSD and MCMD values increases accuracy. A possible explanation for this is that lower MCSD and MCMD values allow cells to subdivide faster yet merge back together with difficulty. As previously discussed with DCS values, the smaller the cells are made, the more accuracy is obtained. Lowering the MCSD and MCMD values have the same effect. However, as with a reduced DCS value, this comes with increased bandwidth costs, as will be analysed next. The scenarios focusing specifically on boundary-optimised situations benefit the most from lower MCSD and MCMD values, as smaller cells more closely represent boundaries (refer to Scenarios 3 and 4). The most interesting results are that the more

realistic scenarios (Scenario 2, Scenario 6, and Scenario 7) show almost no difference in changing the MCSD and MCMD. It is concluded that while the MCSD and MCMD values in theory have an effect on accuracy, in practice the Communication Map algorithms perform well regardless of these settings.

Modifying the MCSD, MCMD, the Minimum Boundary Subdivide Difference and the Minimum Boundary Merge Difference yielded no significant differences to accuracy for scenarios implementing boundaries. The results of all scenarios with boundaries implemented for default and MCSD setting of 2.0 and MCMD setting of 1.0 are shown in Figure 102 to illustrate this.

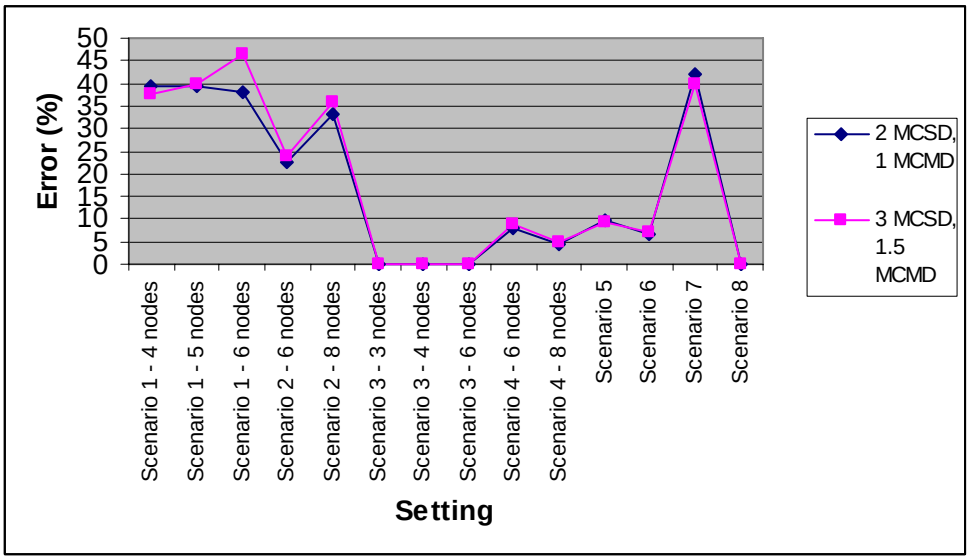

Figure 102: Accuracy of MCSD and MCMD Settings on Boundaries

8.2.8 Bandwidth Requirements

The various settings for the experiments presented in this chapter all have associated bandwidth costs. Bandwidth in wireless networks can be scarce, and lowering bandwidth consumption can be a factor depending on the final environment the CM algorithms will be used. Bandwidth is affected by a number of characteristics, namely:

- how often CMs are broadcast (associated with the duration of each Time Block),
- the quantity of Time Blocks stored by each node and transmitted in each broadcast,
- how large each CM grows, based on the distance nodes travel,

- to what extent each CM is subdivided,
- whether boundaries are implemented or not, and
- the DCS value used.

Two of these characteristics are easily controlled from settings: the use of boundaries, and the DCS value. Figure 103 graphs the bandwidth used for each single broadcast for a single Time Block from a single node given six different settings over four map sizes. This graph shows that a reduction in DCS increases bandwidth dramatically, especially in larger maps. Using boundaries also increases bandwidth by a significant amount compared with its non-boundary counterpart.

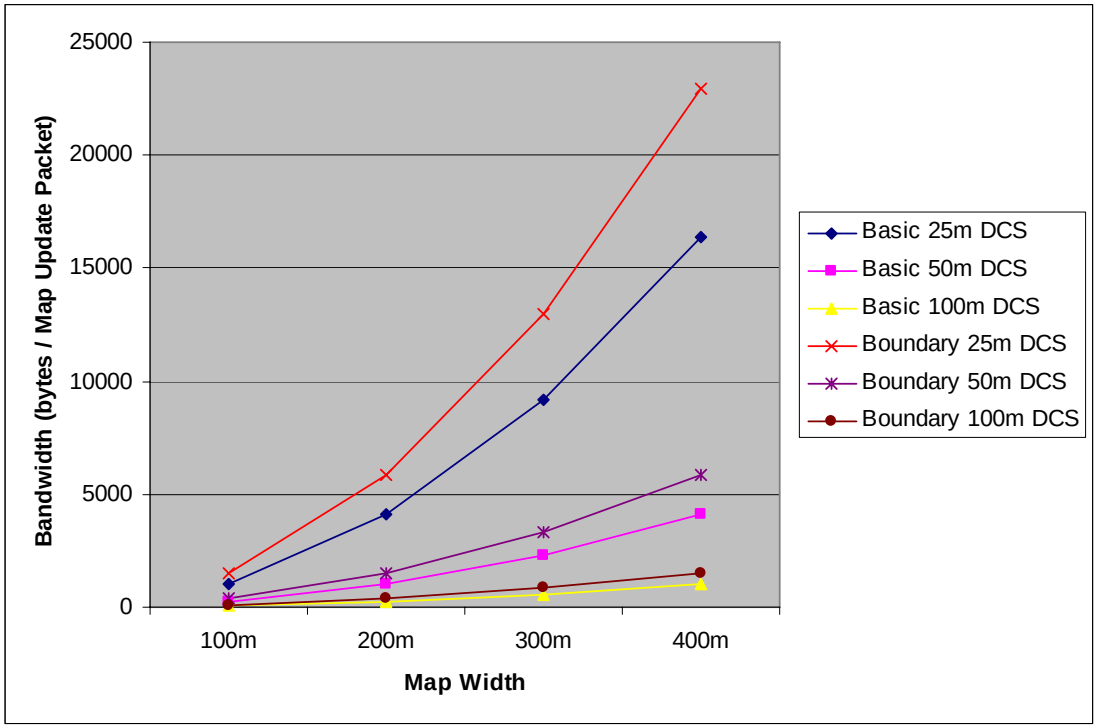

Figure 103: CM Bandwidth Requirements

8.2.9 Summary

An accurate CM is a challenge to construct due to the significant constant coverage of a physical area required. No previous research has attempted to create a signal loss map in real time using only ad hoc nodes. From the scenarios tested in this research it was shown that an accurate CM is difficult to produce. However, nodes in fixed or repeated movement patterns are still able to achieve highly accurate signal loss predictions, despite the actual CM not being identical to the simulated wireless topology.

Accuracy can be improved by reducing the DCS, reducing the MCSD and MCMD, or reducing the duration of Time Blocks. The use of boundaries improves accuracy in specifically-designed scenarios. However, in realistic scenarios the use of boundaries did not show any significant improvement in accuracy, and thus cannot be recommended given the added bandwidth and computational overheads. All of these techniques allow the CM to map more detail, but at a cost of bandwidth. Reducing these values also reduces the CM's ability to average surrounding unsearched territory, and reduces the ability to handle sudden changes in signal propagation. The effect that CM accuracy has on predicting future connectivity, and furthermore on routing protocols, will be studied in the following sections.

8.3 Future Neighbours

The purpose of the FNT is to predict node connectivity. Previous to this research, the only prediction method based on future node movements considered only the current direction and orientation of node travel [Su00]. The FNT algorithms are more complex than this approach, and aim to improve predictions by considering both future node movements and the wireless signal environment.

There are two tests which are used to measure FNT accuracy and two for performance. One measure of accuracy is the percentage of error of a predicted signal loss value recorded for each packet received. The other method is to measure the predicted connectivity accuracy over time. In these scenarios, prediction accuracy was measured at 1 second intervals, where 100% indicates that all predictions for all neighbour connectivity in the network were accurately predicted to exist or not to exist.

Both linear and polling approaches used to generate FNTs are studied. The ratio of cases able to use the linear approach to generate FNTs in relation to the number of cases using the polling approach is measured. The amount of time that each case takes on average to perform is also measured. As one of the suggested advantages of using the linear approach is that it is faster, this performance measurement is used to validate this assertion.

It is difficult to verify the performance of FNT algorithms with only these metrics. Section 8.4, which analyses routing, tests the actual performance of FNT as it can be applied to existing routing protocols. The settings used for these experiments use an ASL of 85 dBm, and recalculate the FNT for all nodes at 1 minute intervals. Boundaries were not implemented in the FNT algorithms as previous results did not show a significant increase in accuracy of the CM by implementing boundaries. Scenario 5 and 7 are the primary scenarios studied, as Scenario 5 was designed specifically to test the FNT, and Scenario 7 is based on a realistic scenario.

8.3.1 Scenario 5

Scenario 5 was specifically designed to test the FNT concept. In this example (refer to Figure 64 earlier) 8 nodes form a network over an area with a large building in the centre. This building has a very high signal loss modifier of 8.0 so that wireless connectivity between nodes will require prediction. The maximum ASL for all nodes is set to 85 dBm. Two of the nodes, C and E, circumnavigate the building a single time. This step maps the signal loss of the building in the CM. Figure 104 to Figure 107 illustrate the connectivity between all nodes as nodes C and E travel along their task paths. Due to the high signal loss of the building, connectivity between several nodes is broken frequently.

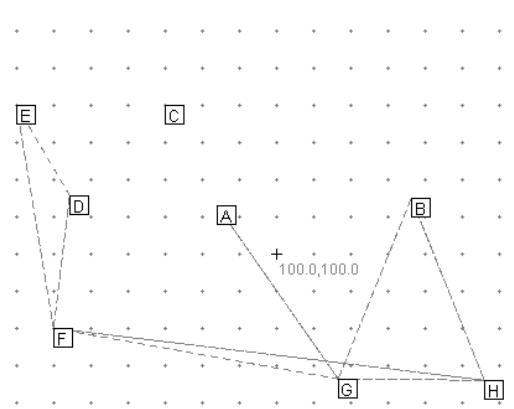

Figure 104: Scenario 5 Connectivity at 0 Seconds

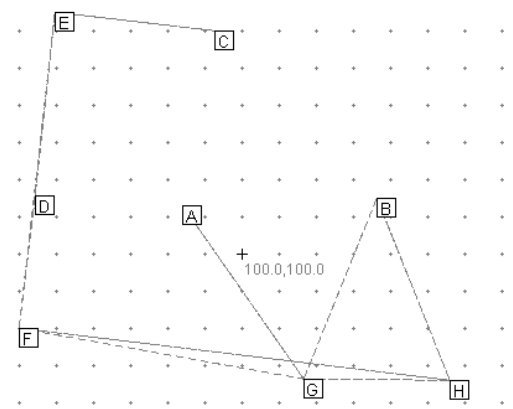

Figure 105: Scenario 5 Connectivity at 30 Seconds

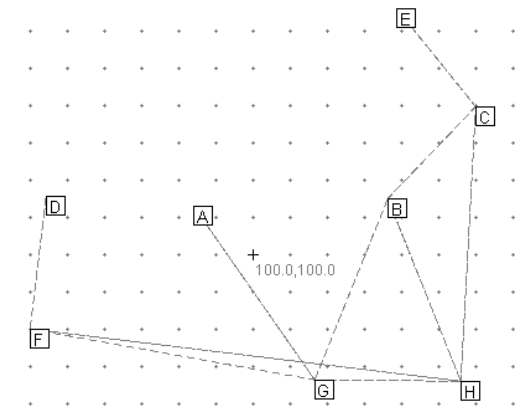

Figure 106: Scenario 5 Connectivity at 180 Seconds

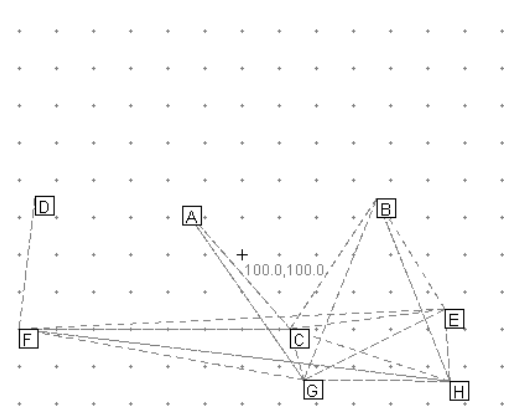

Figure 107: Scenario 5 Connectivity at 300 Seconds

An accurate CM delineating the bounds of the signal loss is created after nodes C and E conclude travel. The CM created by node A at 600 seconds is shown in Figure 108, and due to node map sharing, all nodes have near identical maps. From 600 seconds, Node F traverses the map from left to right, where connectivity between neighbouring nodes will change. The basis of the FNT concept is in predicting these connectivity changes to assist routing protocols.

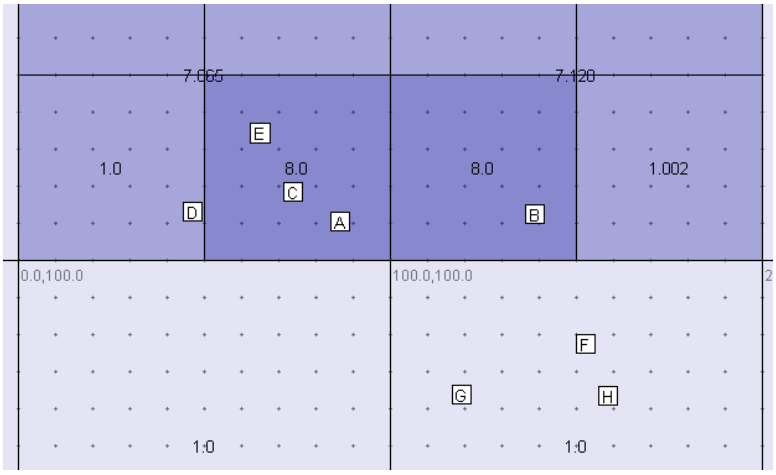

Figure 108: CM of Scenario 5 after 600 seconds

Each node in the network recalculates a FNT for each other node at periodic intervals (every 1 minute in Scenario 5). The FNT's purpose is to list predicted changes in network connectivity based on when signal loss is predicted to fall below or rise above the maximum ASL. The case of nodes A and F is studied. Node A resides in the centre of the map, and thus initially has no connectivity with node F. Node F will travel and pass close enough to node A that connectivity will be established, and then lost as node F moves further away. The FNT generated during this scenario by Node A after the first 60 seconds is shown in Table 8.5. The actual history of connectivity between nodes A and F is listed in Table 8.6. It can be clearly seen that Node A's connectivity is predicted almost perfectly.

Table 8.5: Node A's Predicted Connectivity of Node F

Time	Connectivity Changes
12:36 Neighbours:	true
14:51 Neighbours:	false

Table 8.6: Node A's Actual Neighbour History of Node F

Time	Connectivity
12:36	[ALIVE] Loss: 84.95 dBm
14:50	[UNREACHABLE]

To illustrate the accuracy of predictions for all nodes in this scenario, Figure 109 graphs average predicted connectivity accuracy over time. An accuracy of 100% indicates that all connectivity predictions for all nodes were accurate at that point in time. Three Default Cell Size (DCS) settings are implemented, 25 metres, 50 metres, and 100 metres (the default). This was done to determine how reducing the DCS affects accuracy. DCS is the initial size of each cell when the CM is generated. In theory, the smaller the DCS, the more detail can be represented and thus the more accurately signal loss can be mapped. However, a smaller DCS increases bandwidth during periodic map sharing. With larger

DCS settings, signal loss of smaller interferences is averaged into the larger cells. The results from these three experiments show that once the CM has been created with reasonable accuracy (that is, the signal loss has been mapped correctly), the error margin of predicted connectivity is less than 20%. This is true for all DCS settings used. The default DCS value for this scenario produces marginally better results than lower DCS values. The reason behind this is that with the largest DCS value, the large building is mapped in entirety by default, whereas smaller DCS values take time before relevant cells are between nodes in order to map signal loss to them. It is important to note that a signal loss map cannot represent areas unless those areas have been communicated through. In this scenario the smaller DCS values take nodes marginally more time to discover and attribute loss to each cell as it is communicated through. Results from this experiment show FNT to be accurately able to predict connectivity.

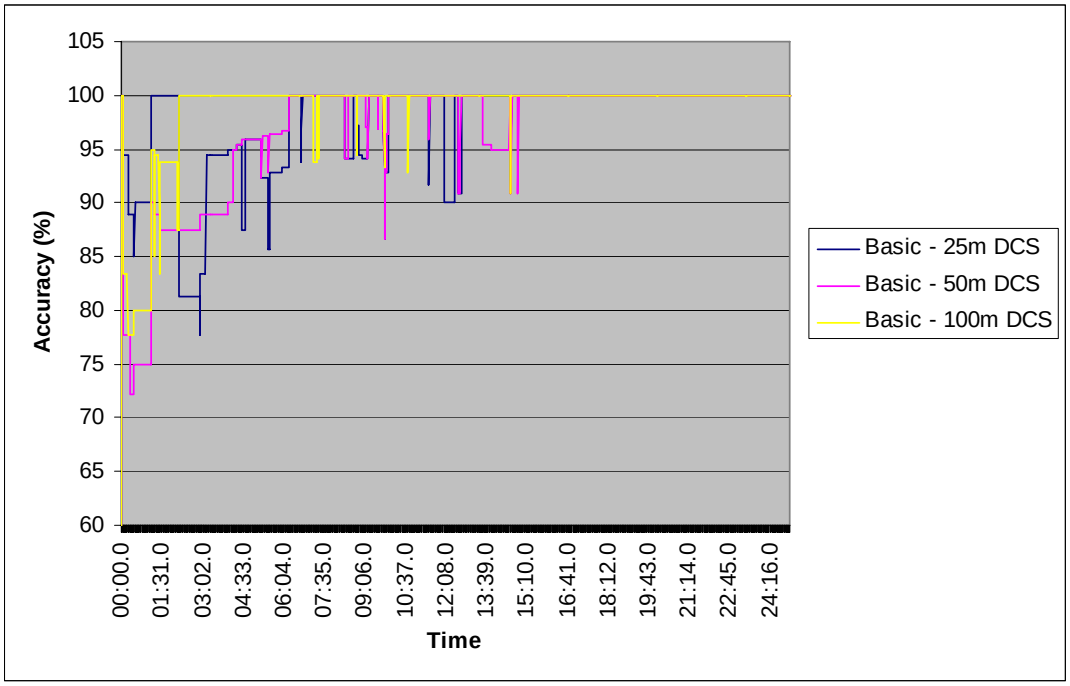

Figure 109: FNT Accuracy for Scenario 5

8.3.2 The Case of Scenario 7

Scenario 7 is based on a real world scenario. In this case the limited number of nodes (6 nodes) travel in only a small part of the simulated map, not enough to cover the complexity of the map (Figure 111). The CM of Node A at the end of the scenario (25 minutes) is shown in Figure 110, and shows that the CM does not accurately represent

the simulated area. However, as stated earlier, it is not possible for nodes to accurately map signal loss over areas that nodes have not visited. Instead the CM algorithms will derive an average signal loss for an area based on received signal loss measurements measured in that same area. As the nodes in Scenario 7 follow a repeated pattern, the accuracy of FNT is still high, regardless of the DCS used (refer to Figure 112). This shows that even in a real world-based scenario where the CM is not entirely known, FNT predictions are within 60% to 100% accurate over almost the entire scenario.

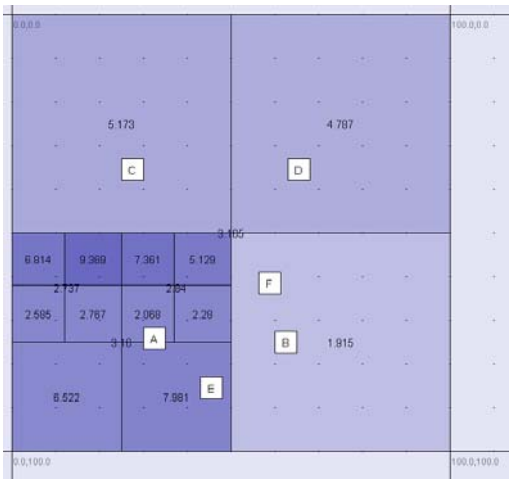

Figure 110: CM of Scenario 7 from Node A

Figure 111: Simulated Wireless Map of Scenario 7

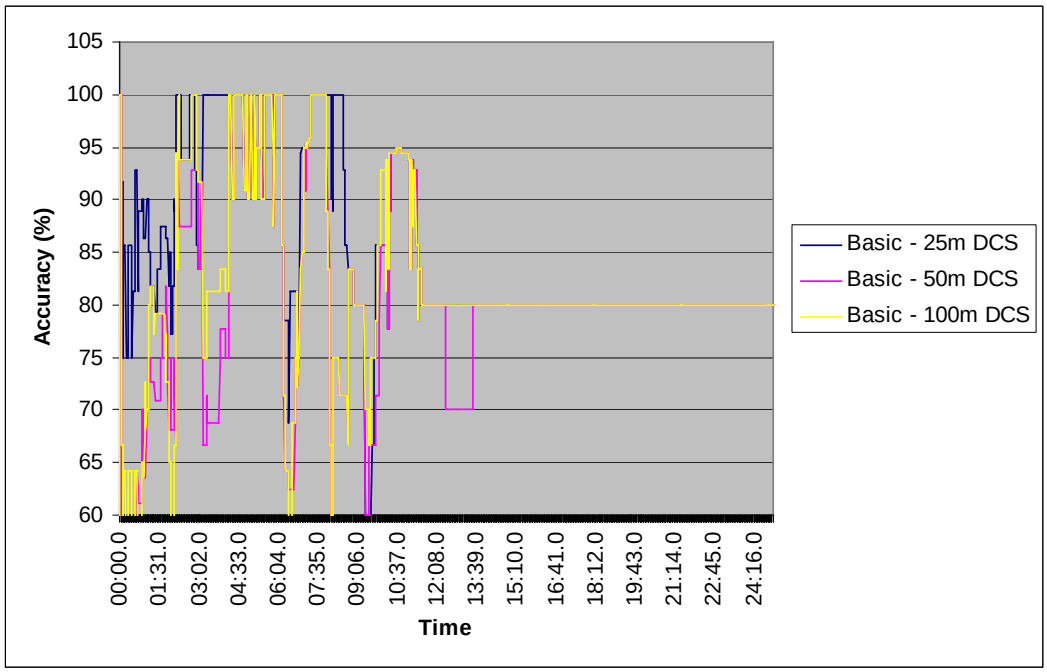

Figure 112: FNT Accuracy for Scenario 7

To further illustrate the accuracy of the FNT during Scenario 7, the connectivity between Node A and Node F is included. Table 8.7 shows the actual connectivity history between Node A and Node F over the entire simulation. Table 8.8 shows the FNT predictions between the two nodes, when calculated at 1 minute and 4 minutes respectively. The two FNTs make similar predictions, however the FNT generated at 4 minutes has a slight improvement in accuracy on all timings due to an improved CM being developed during this time. While the predictions of the FNT are not exact, they are very close, especially considering that the CM developed (Figure 110) was far from a perfect representation of the simulated wireless environment (Figure 111). This illustrates that even in a complex scenario, the FNT has a high level of accuracy. If some form of safety margin is introduced to routing algorithms to ensure routes are handed off earlier than predicted, any small inaccuracies could be avoided entirely.

Table 8.7: Node A's Neighbour History of Node F

Time	Connectivity
0:06	[ALIVE] Signal Loss: 80.896 dBm
0:41	[UNREACHABLE]
1:18	[ALIVE] Signal Loss: 68.764 dBm
2:00	[UNREACHABLE]
3:27	[ALIVE] Signal Loss: 72.405 dBm
5:18	[UNREACHABLE]
7:17	[ALIVE] Signal Loss: 81.870 dBm
10:27	[UNREACHABLE]

Table 8.8: Node A's FNT of Node F

At 1 Minute		At 4 Minutes	
Time	***Connectivity***	***Time***	***Connectivity***
2:34	true		
5:41	false	5:36	false
7:07	true	7:14	true
11:39	false	11:18	false

8.3.3 All Scenarios

The average predicted signal loss accuracy of the FNT across all of Scenarios 1 through 8 (including each node population) over the duration of each scenario is shown in Figure 113. An accuracy of 100% indicates that all connectivity predictions for all nodes were accurate for each second of simulation where predicted connectivity accuracy was polled at 1 second intervals. These are all based on using a CM without boundary implementation, and 100m DCS. The majority of scenarios show very high accuracy of the FNT, regardless of the settings used. To better observe accuracy, the estimated signal loss was compared to the actual signal loss that was predicted as a secondary statistic of the FNT. Figure 114 and Figure 115 show the percentage that predicted signal losses were in error for the various CM settings that were studied in Section 8.2. The same pattern from the CM results in Section 8.2.6 is shown, as FNT signal loss predictions are based on the accuracy of the CM. Reducing the quantity or duration of Time Blocks improves FNT accuracy, as it was previously shown to improve CM accuracy. Similarly, reducing the MCSD and MCMD settings also improves FNT performance.

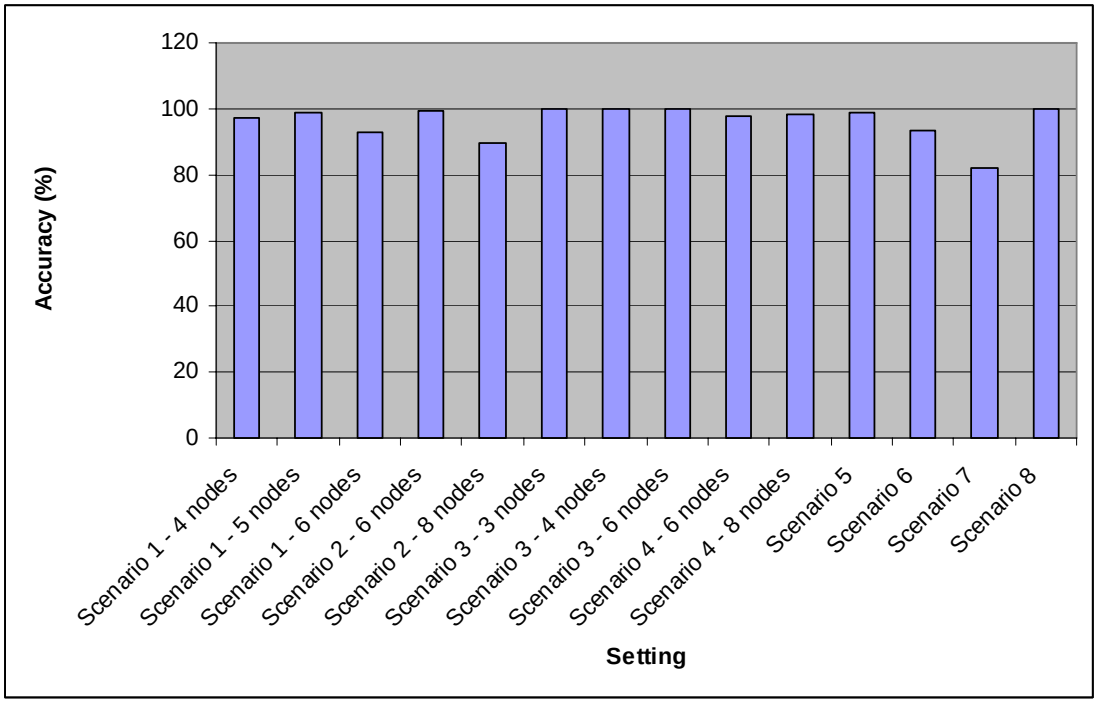

Figure 113: FNT Prediction Accuracy over all Scenarios

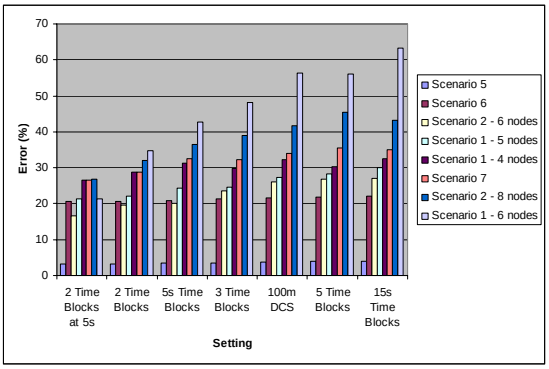

Figure 114: FNT Predicted Signal Loss Error for Time Blocks

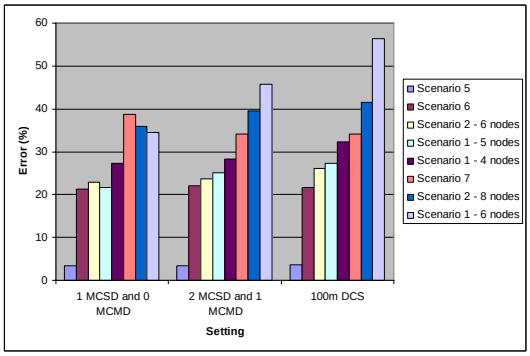

Figure 115: FNT Predicted Signal Loss Error for Subdivide Settings

8.3.4 Computational Efficiency

The efficiency of the FNT algorithms is an important issue due to the two approaches which can be used to calculate the resulting tables. As described in Chapter 5, FNT can be calculated either using linear calculation or by using polling. Due to the complexities of combining task movement and areas of varying signal loss, it is only possible to use linear calculation in some circumstances. Figure 116 shows the proportion of linear cases for all scenarios tested. Less than half of all FNT task segment calculations were able to use linear calculation (Scenario 3 has no moving tasks, and thus has no FNT calculations to make). The argument for using linear calculation over polling is a case of speed vs accuracy. Linear calculation results are more accurate by design. Polling can produce similar results, but at a cost of time. The increase in distance in node tasks will increase the time taken to generate a FNT. In all scenarios, polling used a poll interval of 1 second, as the smallest unit of measure in FNT is 1 second. Figure 117 shows the average time taken for each FNT calculation approach across all scenarios. On average polling takes 42% longer to compute results compared to the linear calculation approach.

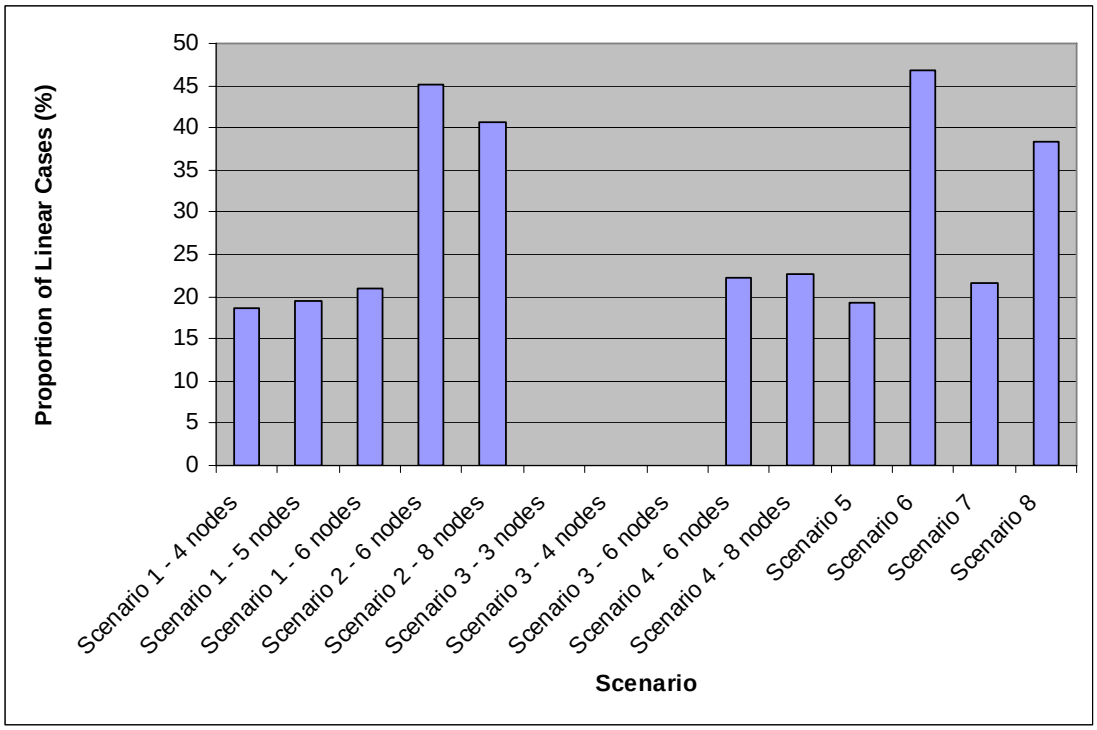

Figure 116: Proportion of Linear FNT Cases for all Scenarios

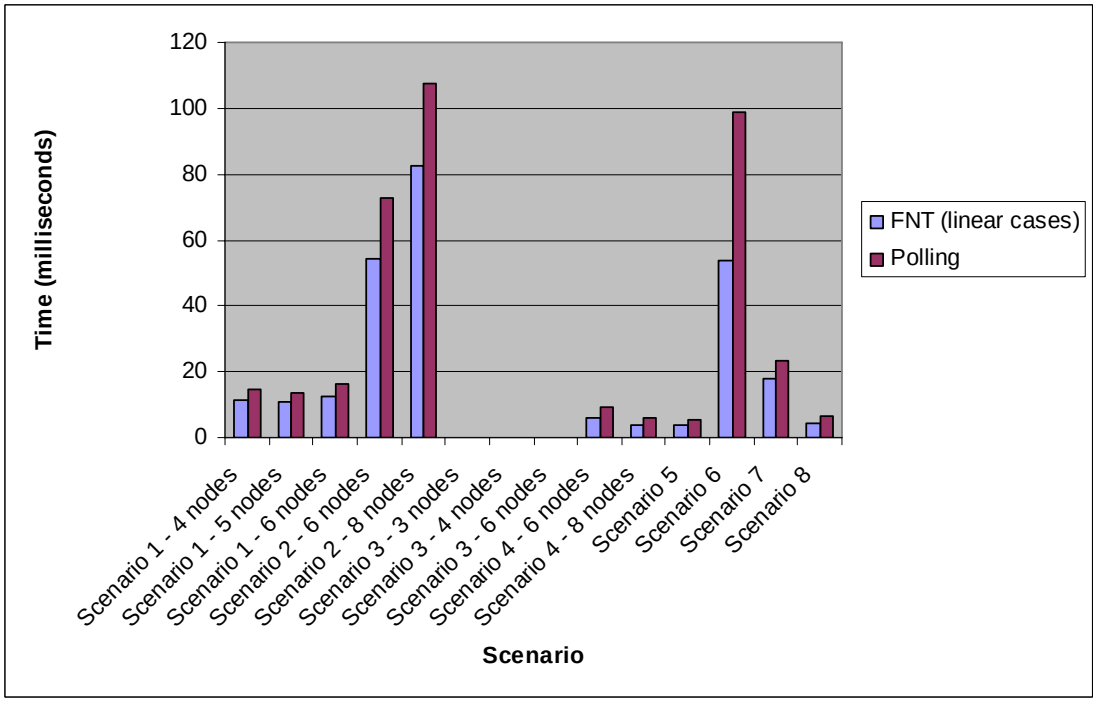

Figure 117: Average FNT Calculation Time for all Scenarios

8.3.5 Summary

The FNT algorithms are able to predict connectivity between nodes with a very high level of accuracy. The accuracy of the task list and CM directly affects the accuracy of FNT predictions. Across all scenarios, regardless of imperfect CMs used, FNT was able to predict most neighbour states. The linear approach to FNT calculations is significantly faster than polling. However, in all scenarios, the use of the polling approach was more frequently required.

The real significance of the FNT and CM algorithms is in their ability to improve connectivity in routing algorithms. The next section studies the performance of FNT and CM across several routing approaches.

8.4 Routing

The CM and FNT have been created to support routing algorithms by providing predictions in connectivity. In theory, the FNT can be applied to any routing protocol to potentially improve performance. This section tests the routing algorithms described in Chapter 6 to analyse the performance of the FNT and the CM on actual routing protocols.

There are two categories of routing protocols studied in this research, on demand and distance vector-based. On demand routing protocols initiate routes as they are required and as they terminate. Entire routes are determined by source nodes requiring routes. In contrast, distance vector routing protocols broadcast routing tables detailing routes to all destinations periodically between nodes. In such networks, source nodes know of only the immediate next hop to a destination, with no knowledge of the route as a whole.

For each category, four primary routing protocols are studied. A traditional routing protocol is studied first, where shortest hop count to a destination is the only route selection mechanism. Such routing protocols make no use of prediction and thus provide a baseline for prediction performance comparisons. Su's routing protocols [Su00] for on demand and distance vector-based routing are studied next, as the only existing prediction-based routing protocols based on future tasks. The CM and FNT designs in this thesis are then compared in relation to Su's algorithms and a traditional approach.

Finally, a modified version of Su's prediction approach is studied, in which signal loss measurements are provided by the CM, instead of the free-space signal loss model.

Further results are studied with the introduction of a safety margin to the FNT algorithms. The safety margin reduces the maximum ASL from which the FNT predicts connectivity. In doing so, routes will be handed off where signal loss is less than the actual predicted loss.

There are several statistics used to analyse and compare the various routing protocols:

- Predicted handoffs;
- prediction accuracy;
- unexpected drop-offs;
- drop-off time without route;
- number of routes established;
- average hop count; and
- network connectivity.

The purpose of these statistics and the aim of this section is to test if the FNT improves connectivity in routing algorithms compared to Su's algorithms. It is hypothesised that Su's algorithms with a CM should perform better than Su's original algorithms that rely on a free-space signal loss model. Furthermore, the use of safety margins in FNT is predicted to further increase connectivity, as routes will be handed over at lower signal loss levels, reducing the chance of CM inaccuracies from mispredicting RETs.

Routing performance is difficult to measure because of the variation in scenarios in which the routing algorithm may be used. As with earlier performance measurements, routing is tested across the eight scenarios with varying node populations. Scenario 3 has no prediction errors in general due to having no moving nodes. Scenario 7 is examined in detail, as it is based on a real life scenario, where much of the signal loss topology is not mapped.

8.4.1 On Demand Routing

On demand routing protocols are analysed primarily as routes are established individually. This permits a more detailed study of route prediction and route handoff than distance vector protocols. The on demand protocols tested are DSR [JM96] (which does not have prediction), FORP [Su00] (as Su's implementation of prediction), a modified version of Su's FORP algorithm that uses the CM, and the FNT + CM designed in this thesis.

The following statistics are gathered to measure the performance of various routing algorithms. Predicted handoffs and prediction accuracy are the primary two as they relate to how accurate route end times are predicted. Predicted handoffs are the percentage of all routes which were predicted to expire before they actually expired, and thus a search for an alternate path could be initiated[1]. Prediction accuracy measures the amount of time that algorithms mispredicted the route expiry time. This presents a more detailed study of prediction performance across algorithms. These two statistics primarily compare the performance of on demand-based routing protocols.

8.4.1.1 Predicted Handoffs

Predicted handoffs is a measure of the percentage of routes which were predicted to terminate before they actually terminated. A predicted handoff occurs whenever a route is predicted to expire and a search for a new route is initiated (whether or not a new route is found is irrelevant). Regardless of how often routes are established, predicted handoffs is a measure of the percentage of these routes which are successfully handed over before being unexpectedly terminated. This statistic identifies prediction performance overall given each prediction algorithm studied. Where a route terminates before it is predicted to terminate (before its RET) is defined as an *unexpected drop-off*, and a route will either be a predicted handoff or an unexpected drop-off. Predicted handoffs measure the accuracy of using prediction to improve connectivity in wireless networks. Connectivity is improved by reducing the number of unexpected route drop-offs, where routes are handed over to more stable routes as they are predicted to expire. Predicted handoffs is a significant performance metric, as predicting route expiration improves connectivity. Each prediction algorithm has a varying number of routes established

[1] Whether an alternate route was found was not measured, as the low node populations of these experiments made such a statistic irrelevant.

overall (discussed in Section 8.4.1.4 below), depending on the frequency of predicted handoffs.

The predicted handoffs for Scenario 7 are presented in Figure 118. Both of Su's algorithms outperform FNT + CM in predicting handoffs. A careful study of routing tables from each of the settings reveals that Su's methods are less susceptible to unexpected drop-offs. Su's algorithms have high accuracy because route predictions are kept short (only the current task's end location is considered), and thus are terminated earlier than using FNT algorithms. While the predicted RETs may be earlier than actual expiry times, handing over routes to more stable ones more frequently does not reduce predicted handoffs. This is evidenced in Figure 120, where it can be seen that both of Su's algorithms have a lower number of unexpected drop-offs. In effect, Su's algorithms are receiving updates at regular intervals as existing routes near expiry. This allows predictions to be updated for changes in the CM. Using FNT, routes have a higher chance of an unexpected drop-off due to the longer duration of FNT predictions. During these longer durations, no updates to routes and RETs are made, due to the nature of on demand protocols. As such, FNT is not shown to improve connectivity over either of Su's algorithms.

Implementing safety margins further reduces predicted handoffs, contrary to what was predicted (Figure 119). Because of the low node population limits, introducing safety margins excludes many routes from being predicted because their signal strength then falls below the acceptable limits. Because the network is not sufficiently populated to have alternate routes to many destinations, many routes are not predicted at all, and consequently have no predicted RETs created (i.e., a RET equal to the very beginning of the route). This results in more unexpected drop-offs. This is the opposite to the prediction that an increase in the safety margin should decrease the unexpected drop-offs and improve the number of predicted handoffs by predicting more stable routes.

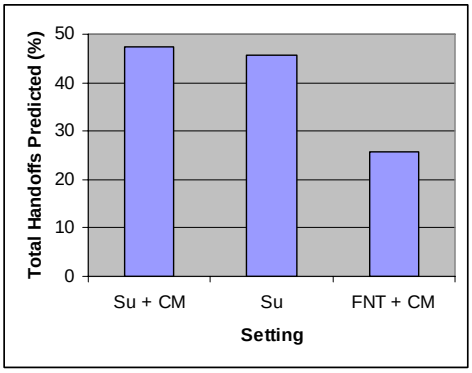

Figure 118: Predicted Handoffs for Scenario 7

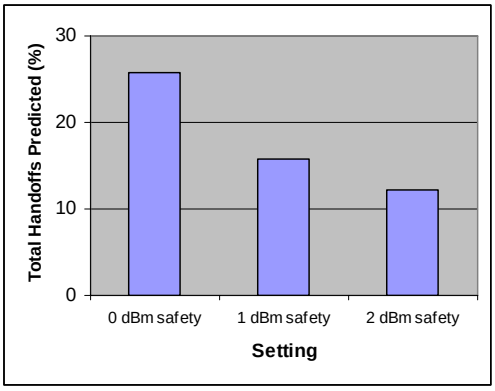

Figure 119: Effect of Safety Margins on Predicted Handoffs for Scenario 7

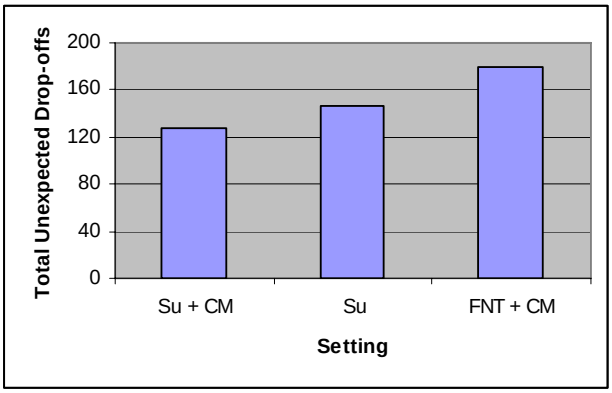

Figure 120: Total Unexpected Drop-offs for Scenario 7

To illustrate where RETs were not predicted, and to what extent predictions were missed, the routing history for Node A's connection to Node F is shown in Table 8.9. This simulation was run using the FNT + CM settings. Nine routes in total were used between Node A and F during the simulation. While the majority of routes were not predicted, the times that they were predicted to terminate were very close to the actual times in most cases. It would thus be reasonable to expect that increasing safety margins would overcome the majority of these cases, provided more nodes existed in the network such that alternate routes were available. Given the complexity of Scenario 7, these results should be similar to those observed in real scenarios.

Table 8.9: Routing History from Node A to Node F

Route	Actual Route Connectivity	Predicted Route Connectivity	Predicted
A - B - F	0:04 - 1:34	0:02 - 1:38	no
A - F	1:34 - 2:00	1:34 - 1:52	predicted
A - E - F	2:04 - 4:21	2:02 - 5:25	no
A - F	4:25 - 5:18	4:23 - 5:46	no
A - B - E - F	5:22 - 5:56	5:20 - 5:59	predicted
A - B - E - F	6:02 - 7:44	6:02 - 7:48	no
A - E - F	7:44 - 8:41	7:44 - 11:21	no
A - F	8:45 - 10:27	8:43 - 11:16	no
A - B - F	11:12 - 25:00	11:12 - 25:00	n / a

The predicted handoff results of Scenario 8 are presented to demonstrate to what extent prediction algorithms perform where signal loss is constant (free-space signal loss) across the entire simulated environment. The predicted handoffs from these experiments (Figure 121) shows much better results for the FNT + CM algorithms. More than 20% of handoffs are predicted for FNT + CM compared with both algorithms based on Su's approach. Both Su's algorithm and the modified Su + CM algorithm perform identically, as with a free-space signal loss model the CM has no effect. A significant portion of the success of FNT + CM is due to the fewer routes established, leading overall to fewer routes which require prediction. The total number of routes established over the duration of Scenario 8 is presented in Figure 122. Because Su's algorithms handoff routes at a change of node direction, significantly more routes are established than using the traditional shortest-hop approach which uses no prediction. Regardless of this, Su's algorithms mispredict 38 route disconnections, compared with only 16 mispredicted by FNT + CM (Figure 123).

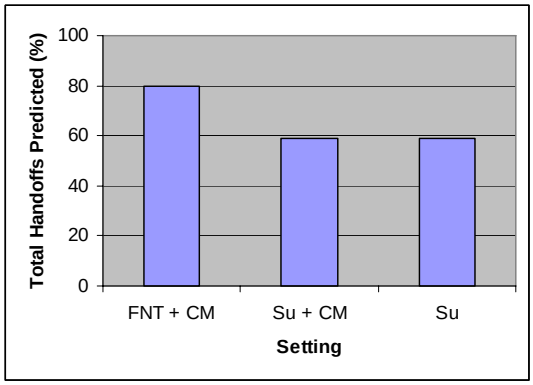

Figure 121: Total Handoffs Predicted for Scenario 8

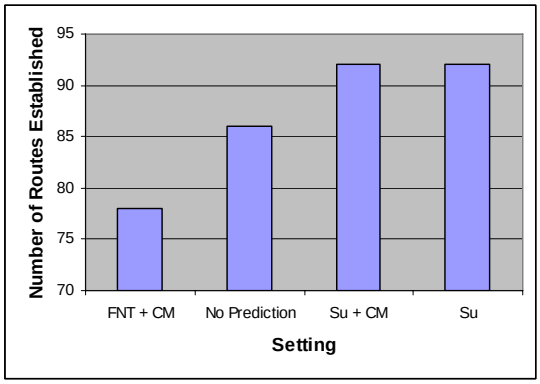

Figure 122: Number of Routes Established for Scenario 8

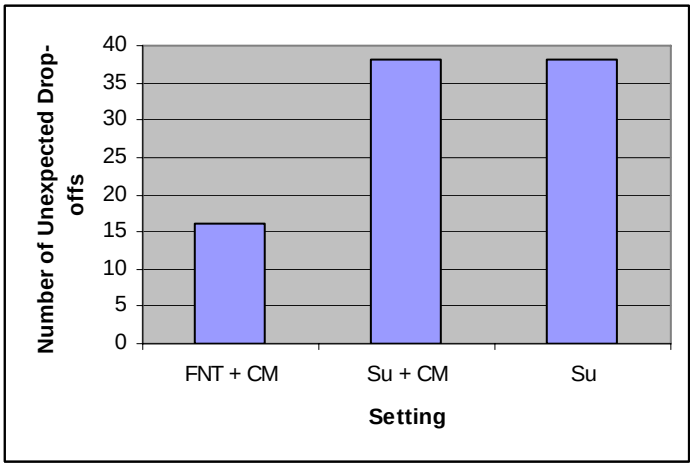

Figure 123: Number of Unexpected Drop-offs for Scenario 8

The total number of handoffs predicted is presented in Figure 124. Both of Su's algorithms perform better overall than the FNT + CM, with the Su + CM algorithm predicting 37% more handoffs than FNT + CM overall. Realistic evaluation of safety margins is not possible due to the low node populations in these scenarios. As such, any prediction algorithm which selects routes to expire earlier than other algorithms has an advantage of predicting more handoffs by minimising the duration that a route is exposed to a potential drop-off. Su's algorithms have such an advantage over FNT + CM. To gain an insight as to how accurate predicted RETs were themselves, the next section studies the relationship between predicted and actual RETs for routes across all scenarios.

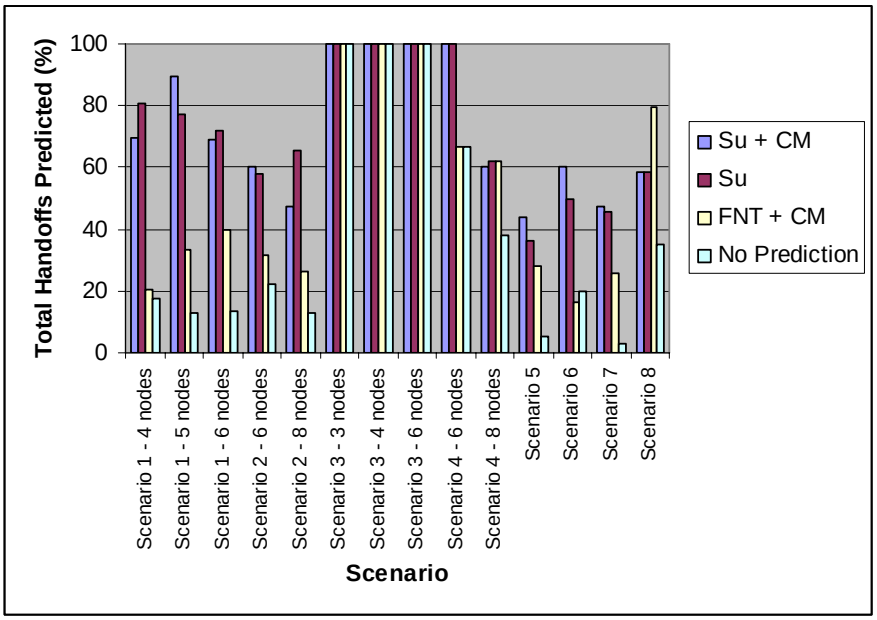

Figure 124: Total Handoffs Predicted for all Scenarios

8.4.1.2 Prediction Accuracy

The prediction accuracy of protocols is a measure of how accurate the predicted RETs were to the actual RETs, measured as the number of seconds between predicted and actual RETs, as a sum total for all routes in a scenario. Figure 125 shows the total prediction error across several settings for Scenario 7. Lower prediction error values are better. FNT + CM performs better than Su's original protocol (FORP), but is slightly outperformed by the Su + CM algorithm. Prediction error of FNT + CM is reduced significantly below Su + CM with the introduction of safety margins, with higher safety margins decreasing prediction error (Figure 126). Figure 127 shows the total prediction error across all scenarios. When the prediction error is averaged across all scenarios, FNT + CM outperforms both Su and Su + CM algorithms. These statistics show that FNT + CM algorithms do predict actual route expiry times better than both Su's methods, although Su + CM performs very well.

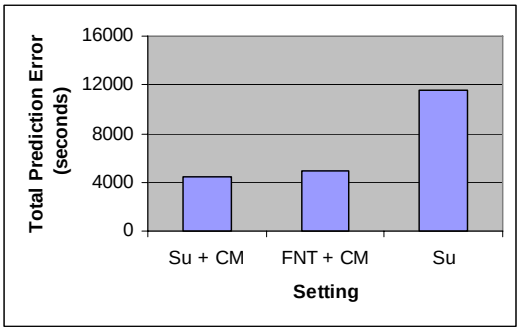

Figure 125: Total Prediction Error for Scenario 7

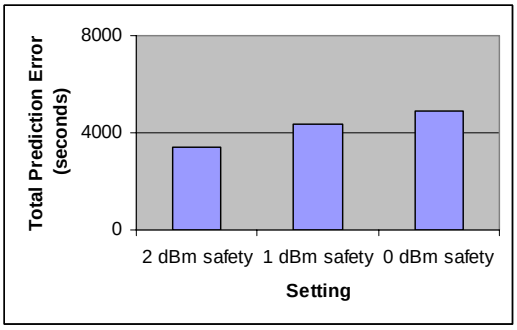

Figure 126: Effect of Safety Margins on Total Prediction Error for Scenario 7

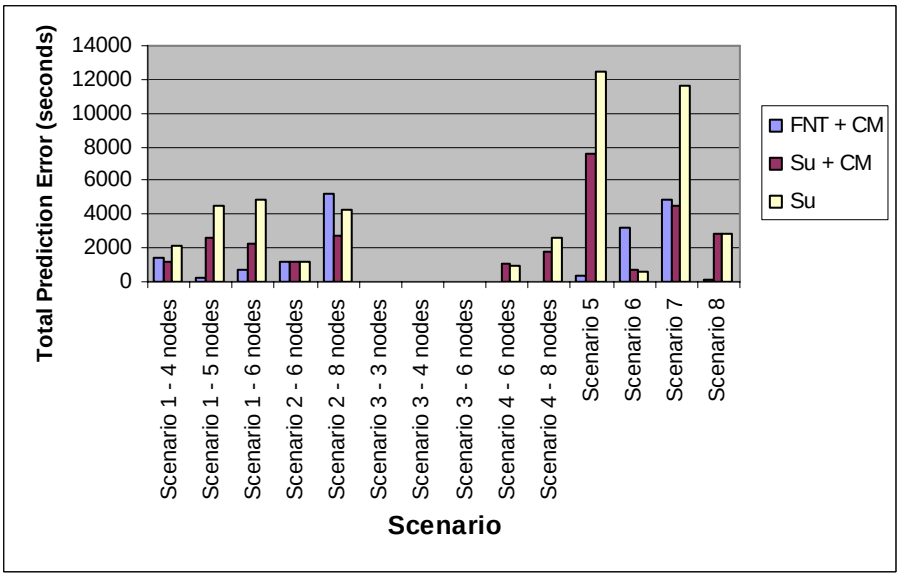

Figure 127: Total Prediction Error over all Scenarios

8.4.1.3 Drop-off Time Without Route

In addition to route stability through predicting handoffs, network connectivity as a whole is also an important performance metric. Network connectivity is reduced by unexpected drop-offs, as finding a new route to a destination after a drop-off requires time. Redundant routes were not added to the routing algorithms. For each route that is unexpectedly terminated, it takes a period of time to realise the route no longer exists, and further time to search for a new route. In these scenarios, the time before a link is declared unavailable is set at 2 seconds. For the on demand routing protocol selected, the destination will wait 2 seconds collecting route requests arriving from the source via different paths before it selects the best route, which is then propagated back to the source. Each hop for any broadcast is at a rate of 1 cycle per hop (a cycle is set to 100ms

in all scenarios). As can be seen from Figure 128, Figure 129, and Figure 130, the time without a route closely follows the predicted handoff results presented earlier, with both Su's algorithms performing better than FNT + CM. It is also observed that increasing safety margins increases the time without route (Figure 129).

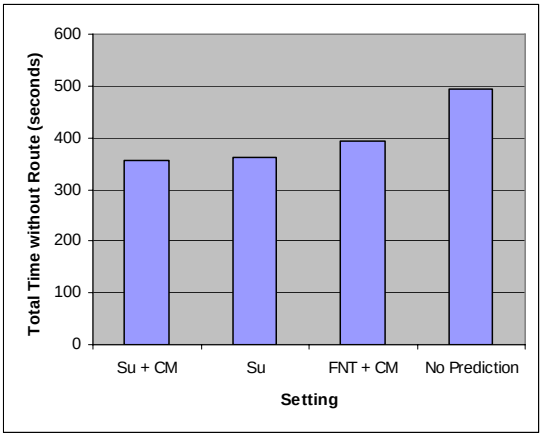

Figure 128: Total Time without Route due to Drop-off for Scenario 7

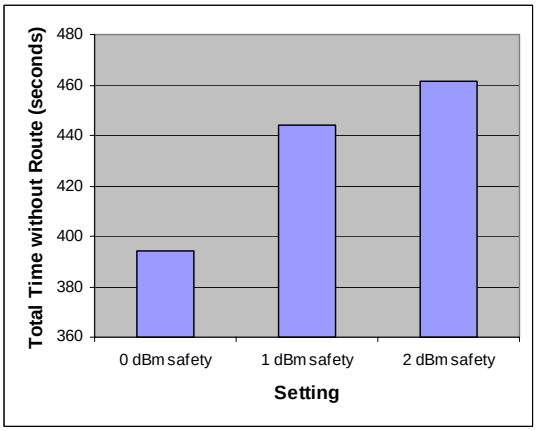

Figure 129: Effect of Safety Margin on Total Time without Route for Scenario 7

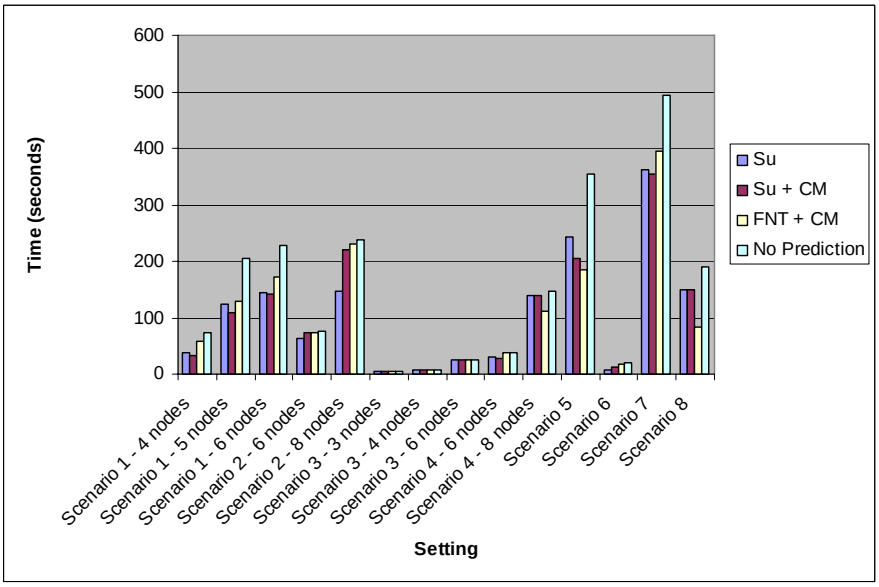

Figure 130: Total Time without Route due to Drop-off for all Scenarios

8.4.1.4 Number of Routes Established

The number of routes established is another important metric, as an increase in the number of routes established also increases bandwidth overhead. Figure 131 shows the total number of routes established for Scenario 7. It is shown that the number of routes between all on-demand routing settings is between 200 and 275 hops for the entire

simulation, all reasonably similar to each other. The lowest number of routes established occurs when using no prediction at all, where route selection is based only on a shortest-path metric. Because these settings do not handover routes to increase stability, they maximise time on each route and thus a lower number of routes are established. FNT + CM produces fewer routes than both of Su's algorithms, as routes are selected based on route longevity. Su's algorithms keep route durations short, as RETs are limited to the current task that nodes are executing. The number of routes established for all simulations is graphed in ascending order in Figure 132, and shows the same trend as that of Figure 131. These results show that FNT + CM minimises the number of routes established, although generating more than using no prediction in order to improve route stability.

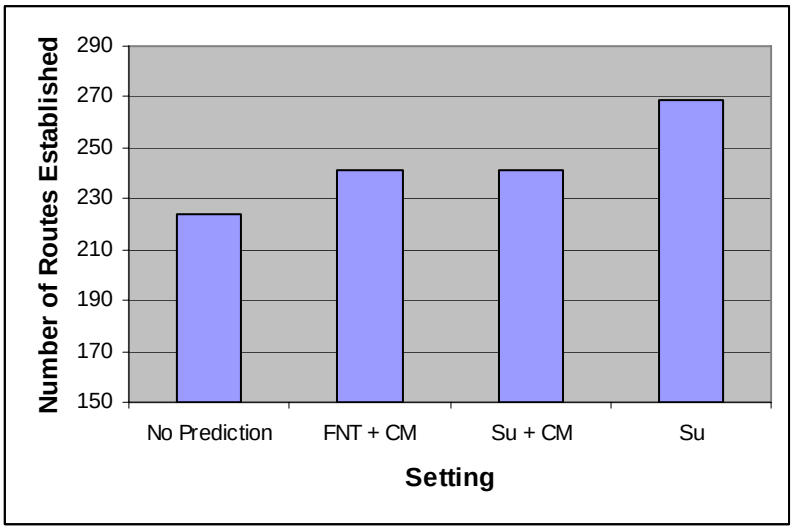

Figure 131: Routes Established for Scenario 7

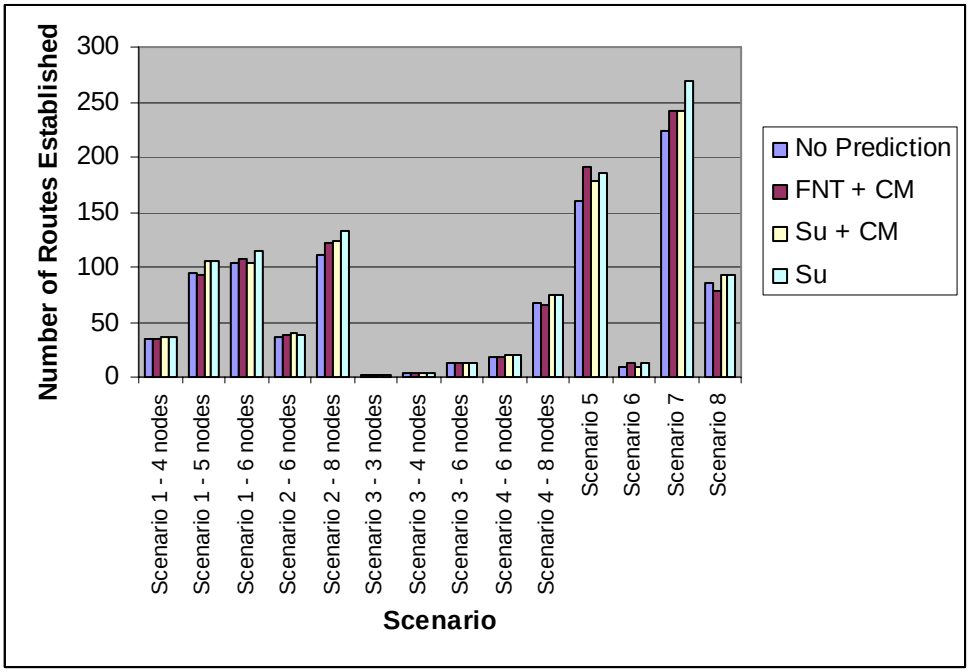

Figure 132: Routes Established for all Scenarios

8.4.1.5 Average Hop Count

The average hop count for a scenario indicates the average number of nodes in each route. An increase in hops along a route's path increases the latency of transmitted packets between source and destination. Both the FNT and Su's algorithms are designed to increase stability, possibly at the expense of longer routes. Both prediction algorithms use the lower hop count route as a secondary selection criteria where two routes are equally stable. Figure 133 details the average hop count for Scenario 7. Note that all scenarios tested in this thesis have very similar average hop counts, between 2.04 and 2.08 for Scenario 7, and between 1 and 2.5 over all scenarios (Figure 134). These low differences are due to the node population restrictions of the simulator. However, a trend between settings is still observable. The lowest average hop count is obtained by using no prediction at all. Su + CM generates the highest average hop count, though as shown earlier Su + CM achieves the greatest number of predicted handoffs, and this is the primary goal of using prediction in wireless ad hoc networks. Figure 134 illustrates the average hop count in ascending order for all scenarios, with FNT + CM generating a slightly higher average hop count over all scenarios. Average hop count is shown to increase where predicted handoffs also increase, showing that route stability comes at a cost of hop count.

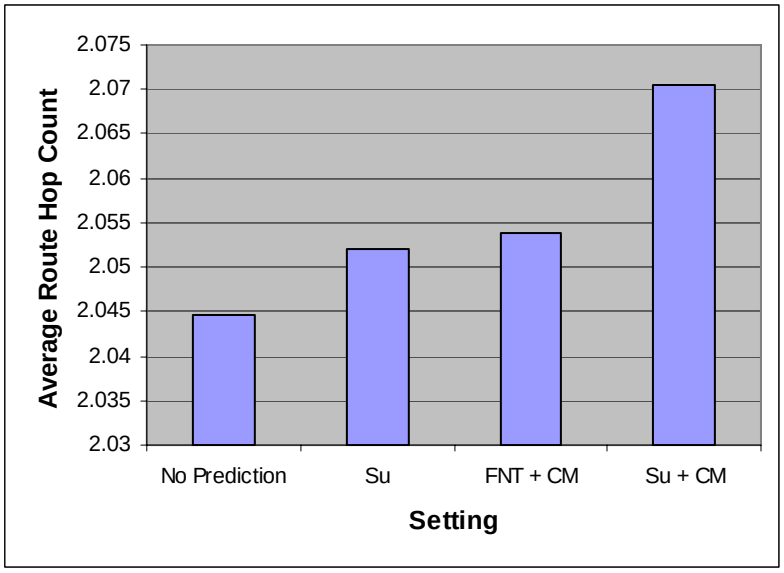

Figure 133: Average Hop Count for Scenario 7

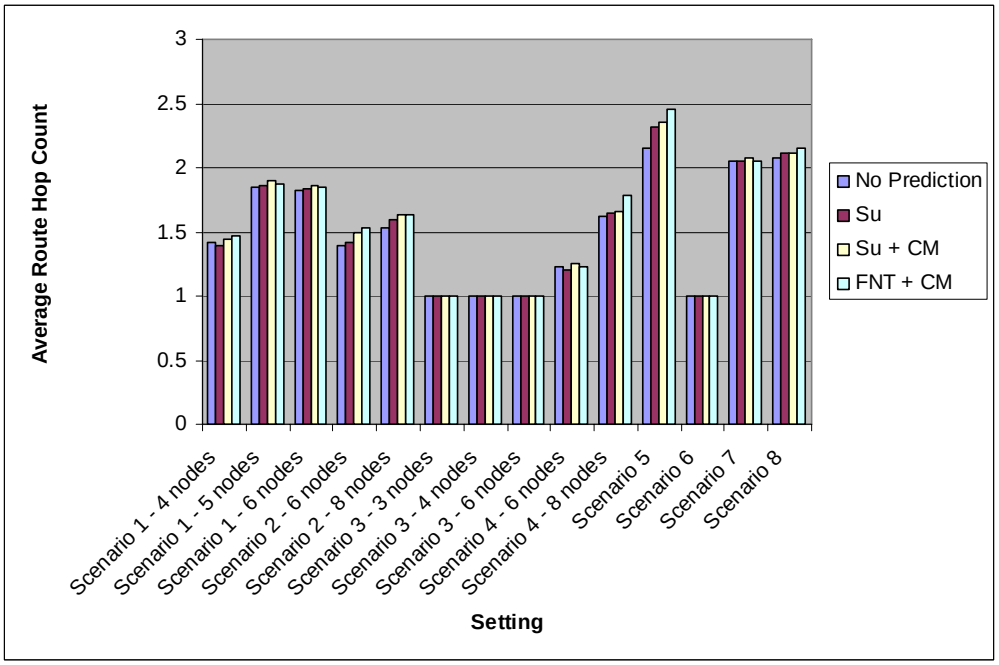

Figure 134: Average Hop Count for all Scenarios

8.4.1.6 Network Connectivity

As a matter of interest, the overall connectivity for each scenario is included in Figure 135. Connectivity is calculated as the percentage of time that each node has a route to every other node. It does not specifically reflect the performance of any particular prediction algorithm, although each algorithm is presented for interest. Results are averaged for each setting and each scenario, grouped together in scenarios. As can be

seen, most scenarios had less than 50% connectivity, due to the low node density. An increase in node population does not always increase connectivity because sparsely separated nodes have fewer connection possibilities. Thus, a small increase in node population, such as in Scenario 1 increasing from 5 to 6 nodes, or in Scenario 2 increasing from 6 to 8 nodes, decreases connectivity.

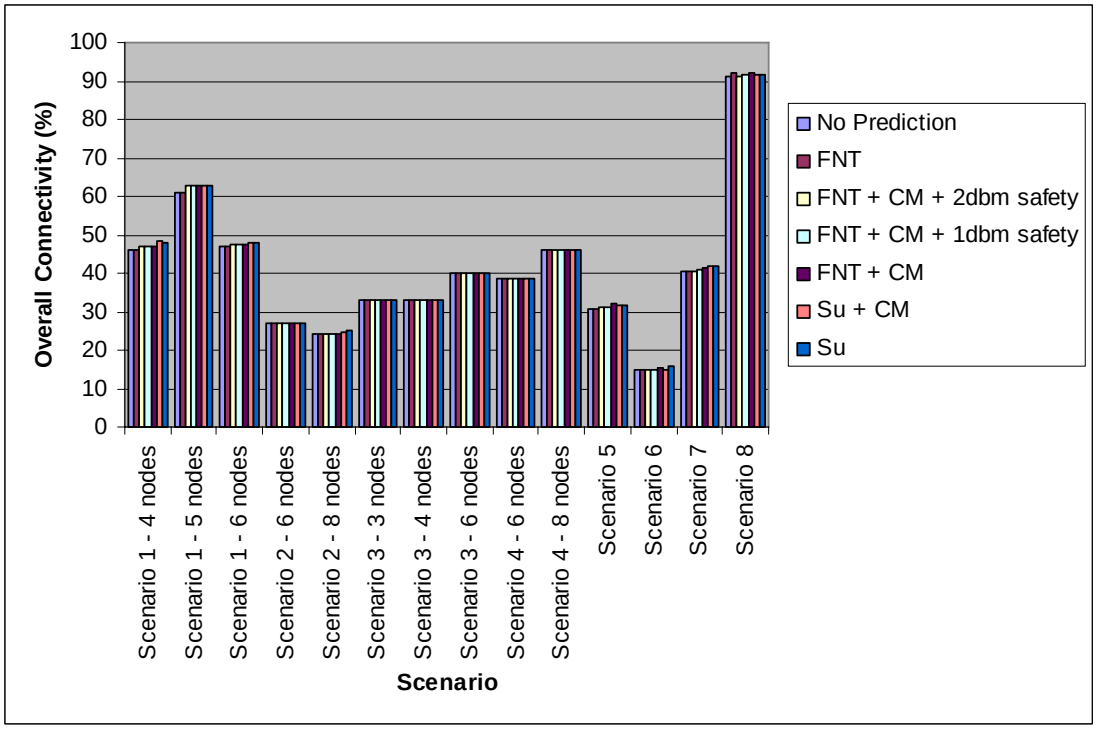

Figure 135: Overall connectivity

A detailed look at connectivity for Scenario 7 is shown in Figure 136. The trend shows both of Su's methods provide better connectivity than FNT + CM overall. This same trend was observed during analysis of predicted handoffs, where both of Su's methods handover routes earlier and avoid many unexpected drop-offs. All prediction algorithms outperform using no prediction.

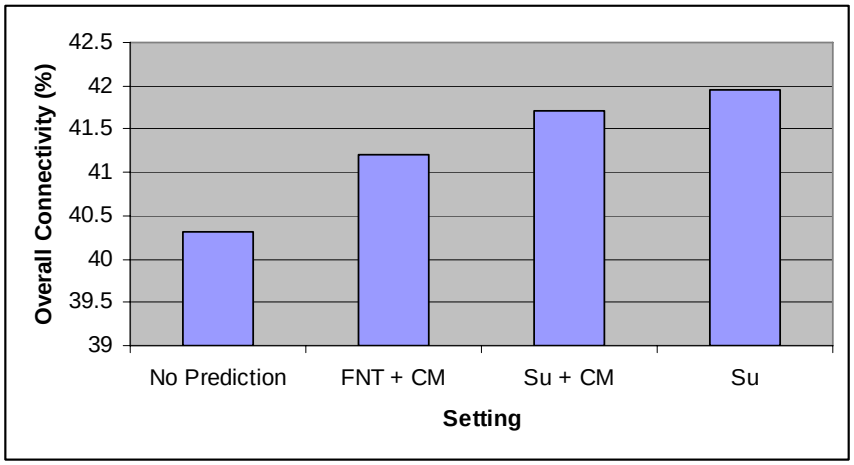

Figure 136: Overall Connectivity for Scenario 7

8.4.1.7 Relation Between CM Accuracy and Route Predicted Handoff Accuracy

The performance of FNT is dependent on the accuracy of the CM. To verify this, Figure 137 was created to show a scatter plot of the average CM error compared with the number of drop-offs for each scenario, implementing both FNT and CM. A weak correlation coefficient of 0.2965 was found and the line of regression is graphed to illustrate that an increase in CM error adversely affects the number of drop-offs in a given scenario. There are two important points to take into consideration in this graph. Firstly, Scenario 7 and Scenario 5 have a significantly higher drop-off count than other scenarios, due to the fact that both scenarios have the most detailed movement patterns where extremely high simulated signal loss areas cause node communication to be cut completely in some areas. Secondly, all three instances of Scenario 3 had no node movement, and as such had no drop-offs as signal loss did not change. The remaining scenarios show on average that as CM accuracy improves, drop-offs decrease.

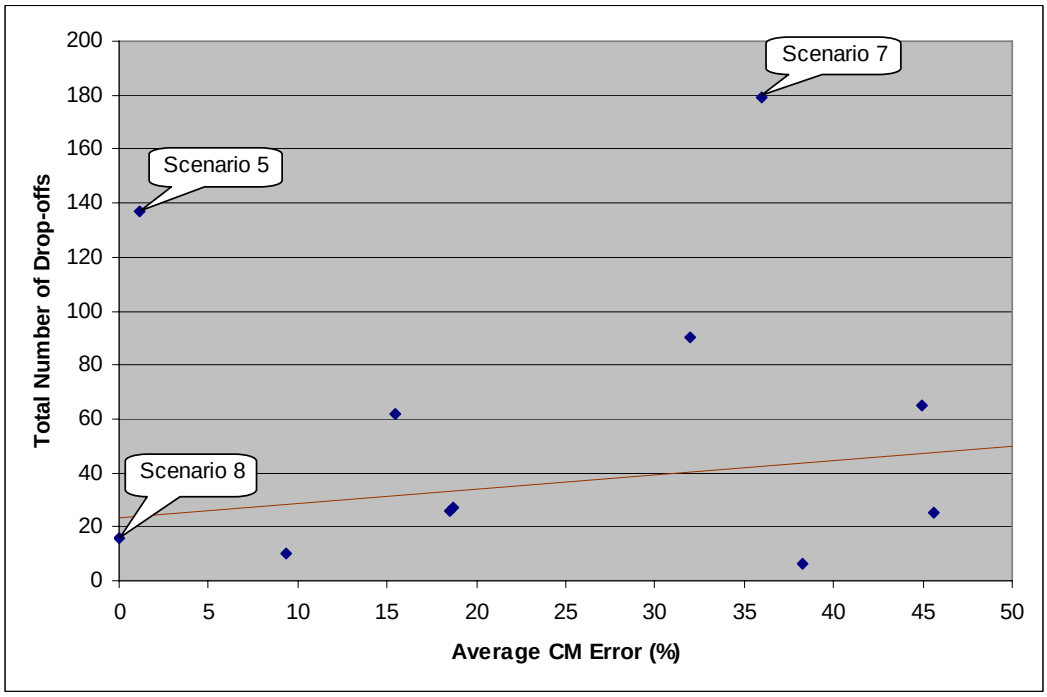

Figure 137: CM Accuracy compared with Route Drop-offs

Also studied is the effect of the CM on the FNT. The FNT is used both with and without the CM, and compared alongside using no prediction. Scenario 7 is studied as it has the greatest reliance on a CM given its varied signal propagation characteristics. Figure 138 shows that using a FNT without a CM was not significantly different in predicting handoffs over using no prediction at all. Figure 139 further shows the number of routes established over the three variations in settings. Using a FNT alone reduces the number of routes established over using no prediction, as routes are selected for longevity. Without the CM however, these routes are mostly terminated unexpectedly. FNT + CM increases the routes established significantly, in order to continually select more stable routes. The importance of the CM to the FNT is therefore important.

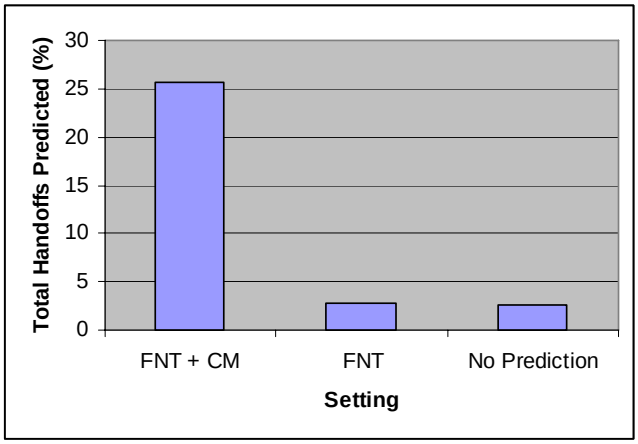

Figure 138: Handoffs Predicted for Scenario 7

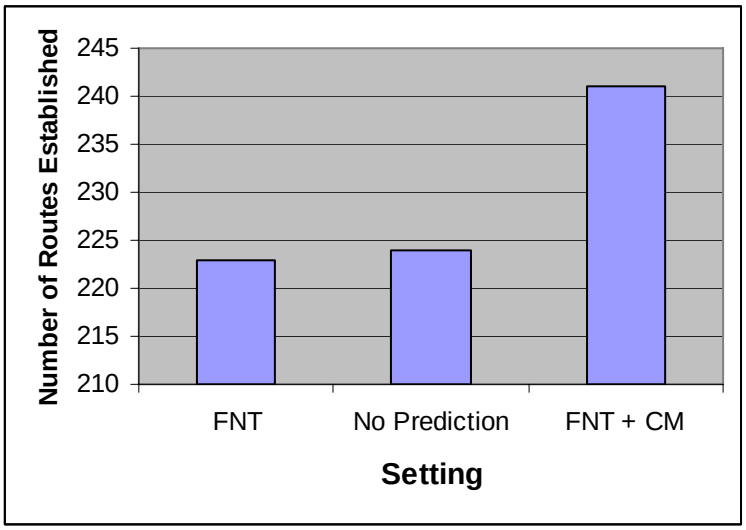

Figure 139: Total Number of Routes Established for Scenario 7

8.4.2 Distance Vector Routing

Distance vector-based routing algorithms differ significantly from on-demand-based routing algorithms in that routing tables are broadcast periodically between nodes, instead of being initiated by each source node. This results in routes being unavailable between a route drop-off and the next route being discovered by a routing table broadcast. However, one of the primary benefits of such approaches is that RET values are updated frequently, improving handoff accuracy of established routes. The base distance vector routing algorithm used for these simulations is DSDV [PB94], and Su's [Su00] distance vector implementation used is DV-MP. In the implementations for this thesis, routing tables contain only the best route to a destination, with no redundant

routes. The broadcast interval is set to 10 seconds by default, with 20 second and 30 second values also tested.

Three performance statistics are used to compare prediction algorithms in distance vector routing. These statistics are also compared with their on demand counterparts to evaluate performance between the two approaches to routing. Unexpected drop-offs are measured to verify prediction performance across the various solutions. The duration of total connectivity is studied to compare how on demand protocols and distance vector protocols ensure network connectivity overall. The number of routes established is detailed to show how routing is performed in general over the different approaches.

8.4.2.1 Unexpected Drop-offs

The number of unexpected drop-offs measures the total number of routes during an entire scenario that were disconnected before they were predicted to expire. The number of unexpected drop-offs for Scenario 7 is presented in ascending order in Figure 140. Both distance vector-based and on demand-based results are included. The figures show that there are fewer drop-offs in distance vector-based protocols for all but Su's algorithm with a CM. Because routing tables are broadcast periodically, nodes receive updated routes at regular intervals. This has a similar effect to Su's algorithms when using on demand routing, where a shorter predicted RET reduces the probability that a drop-off will occur during that time. Both of Su's algorithms perform better than FNT + CM, with Su's method without using a CM performing best. However the CM proves to be a slight hindrance in Scenario 7 for Su + CM as it selects several more routes than Su's algorithm, 2 more of which are unexpectedly terminated. Figure 142 shows the total unexpected drop-offs across all scenarios, verifying the trend shown in Scenario 7 specifically.

In Figure 141, increasing the broadcast time reduces the number of drop-offs, the reverse of what would be expected. This is because fewer routes are established, and thus fewer which could potentially terminate unexpectedly. In distance vector protocols, routes are only re-established after future route broadcasts are received. This does not necessarily imply that increasing broadcast time increases the stability and connectivity of the network, as will be studied further with scenario connectivity next. As unexpected drop-offs are not the only measure of performance, connectivity will be studied next.

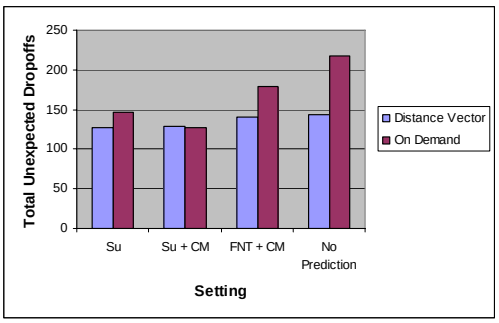

Figure 140: Unexpected Drop-offs for Scenario 7

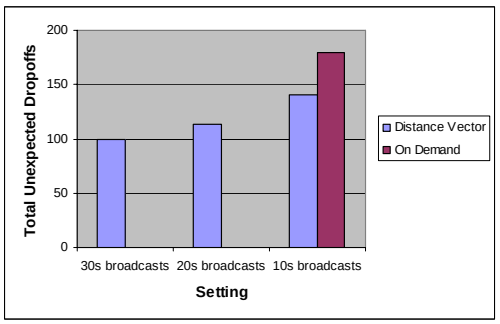

Figure 141: Effect of Broadcast Interval on Unexpected Drop-offs in Scenario 7

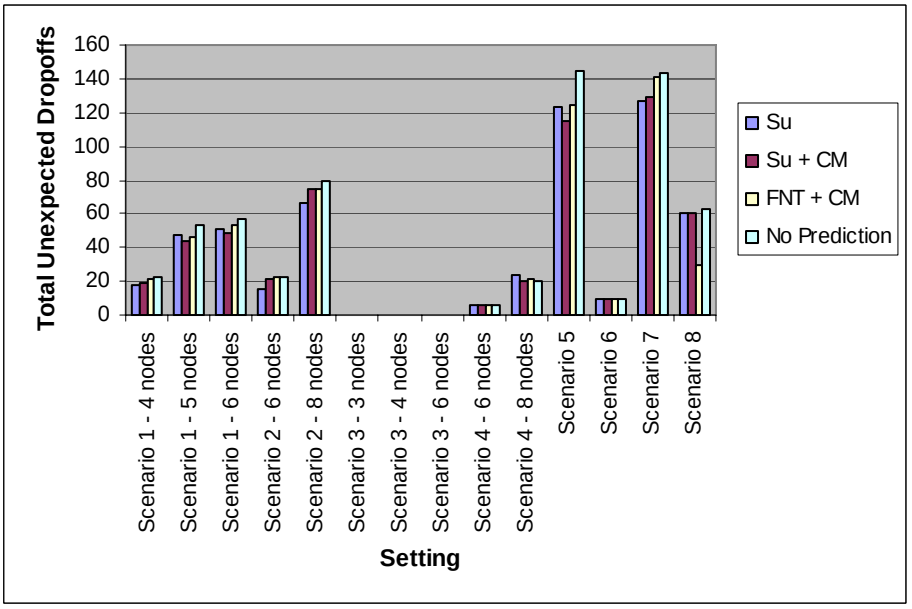

Figure 142: Unexpected Drop-offs for all Scenarios

8.4.2.2 Connectivity

In distance vector-based routing algorithms there is no time lost due to unexpected drop-offs. When a route fails, a node waits until a new route is advertised in a routing table broadcast before the route is re-established. Thus connectivity is studied instead. The connectivity for each of the settings used for Scenario 7 is shown in Figure 143, which graphs the amount of time spent connected with all routes in a simulation as a sum total. The connectivity of on demand routing results is shown alongside. Connectivity is reduced in distance vector-based protocols in all cases because nodes do not receive a new route after an unexpected drop-off until the next routing update (alternative routes are not kept). This is evidenced by the fact that increasing the broadcast time reduces

connectivity (Figure 145). Connectivity is very similar across all algorithms, though FNT + CM has the least connectivity of all.

Figure 144 shows the effect of safety margins on connectivity. Connectivity for Scenario 7 was unaffected by the introduction of safety margins, as the smaller routes detected in on demand-based protocols where no safety margin was used expired before distance vector broadcasts propagated throughout the network.

The connectivity for all scenarios is shown in Figure 146, where the worst connectivity comes from using no prediction at all. Without significantly larger node populations, distance vector-based protocols are significantly outperformed by their on demand counterparts. In the node populations tested in Scenarios 1 through 8, in most cases there are no alternative paths from which distance vector-based protocols can use.

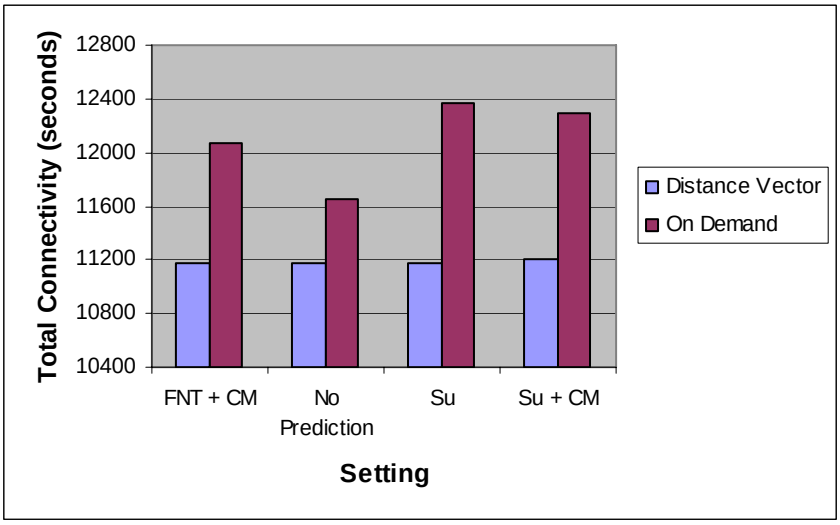

Figure 143: Connectivity for Scenario 7

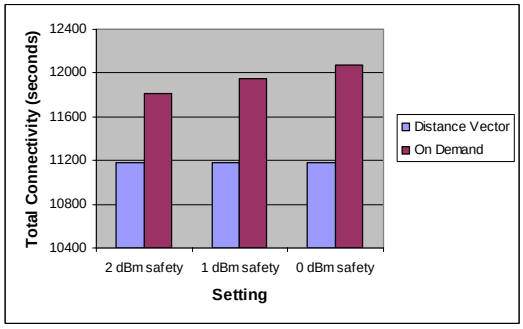

Figure 144: Effect of Safety Margin on Connectivity in Scenario 7

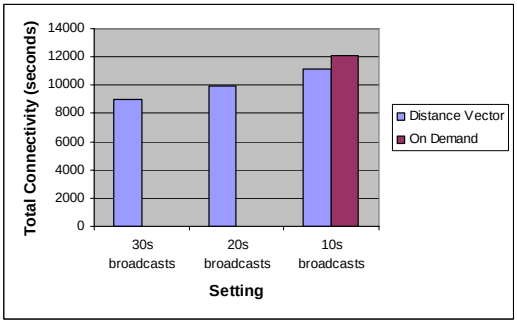

Figure 145: Effect of Broadcast Interval on Connectivity in Scenario 7

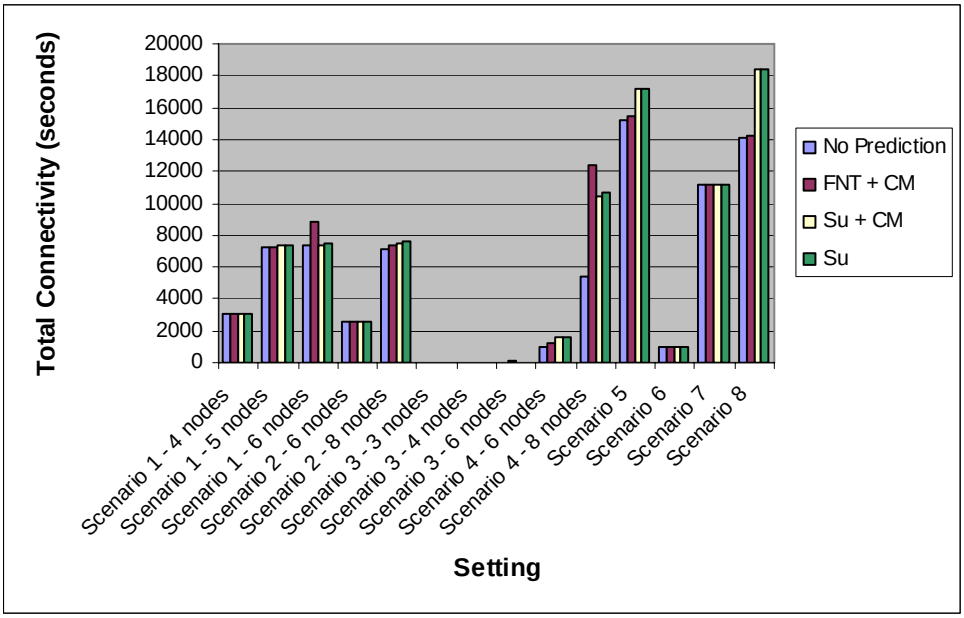

Figure 146: Connectivity for all Scenarios

8.4.2.3 Routes Established

The number of routes established is shown for Scenario 7 in Figure 147. The fewest routes in almost all cases come from using the on demand-based routing approach. On demand protocols only establish a new route when an existing route is broken, unlike distance vector protocols where routes can be re-established with every broadcast, if a better route is made available. Consequently, the more frequent the routing broadcasts, the greater the increase in route establishments (as shown in Figure 149). Similarly, an increase in safety margin reduces the number of routes, as fewer routes are predicted (Figure 148). In Scenario 7, Su's method with a CM causes even more route establishments than the FNT approach, due to Su's method frequently selecting better routes.

The number of routes established for all scenarios is shown in Figure 150. FNT + CM establishes more routes than other algorithms, though this is due mostly to Scenario 4 using a population of 8 nodes. In the field of wireless ad hoc networking it is difficult to propose a single algorithm to perform consistently better across all scenarios, due to the variety of routing scenarios, node mobility patterns, and varying signal propagation.

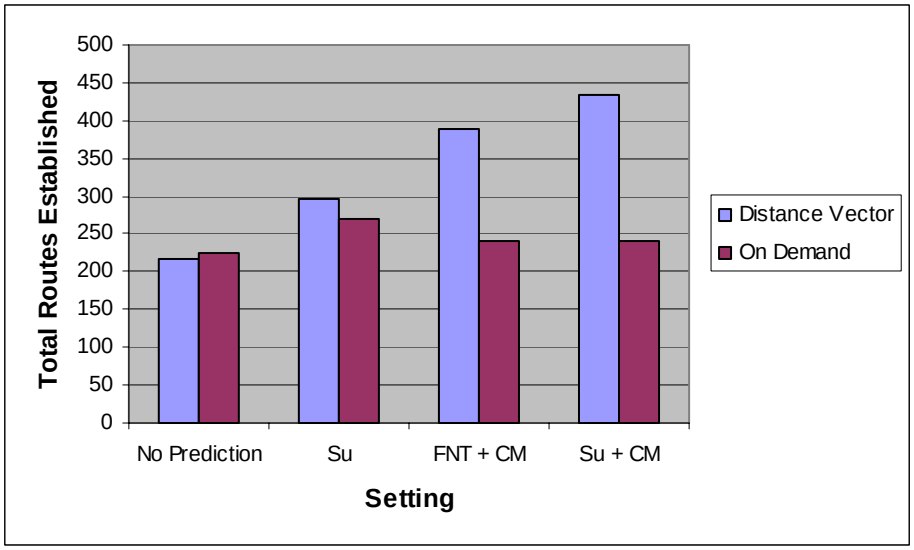

Figure 147: Routes Established for Scenario 7

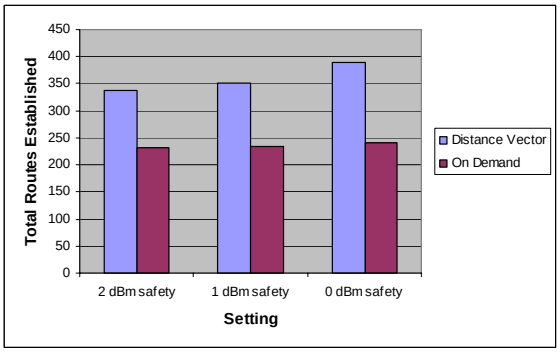

Figure 148: Effect of Safety Margin on Routes Established in Scenario 7

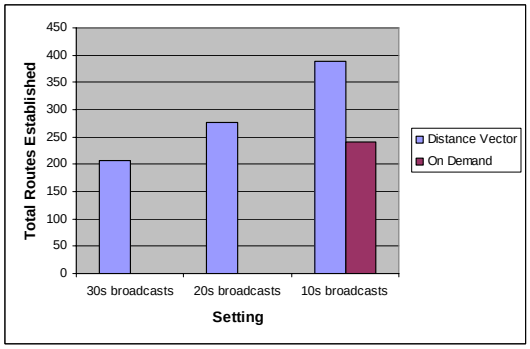

Figure 149: Effect of Broadcast Interval on Routes Established in Scenario 7

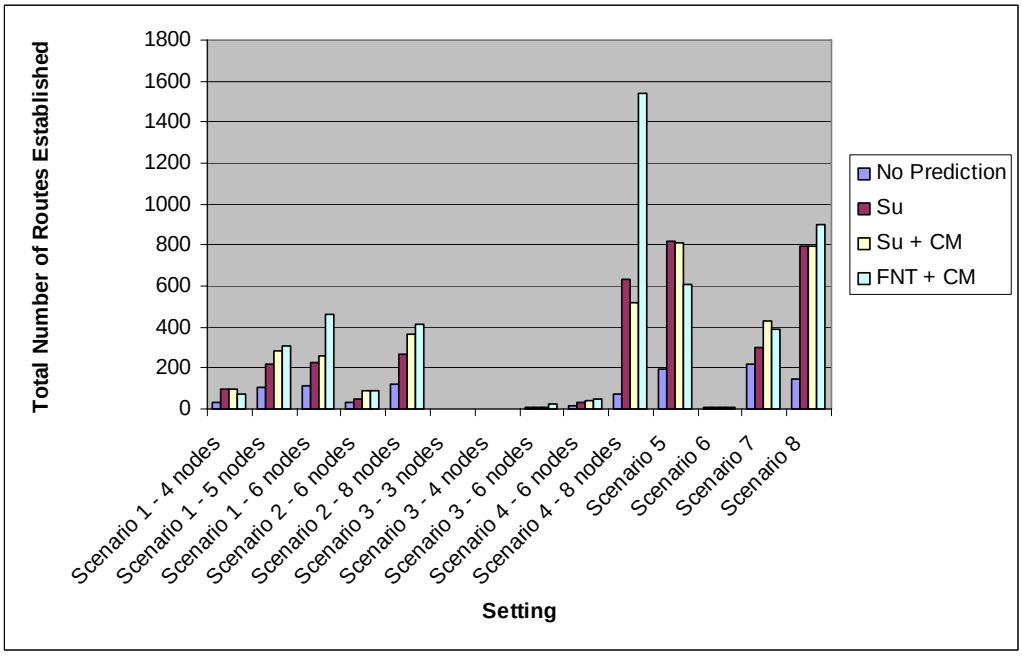

Figure 150: Routes Established for all Scenarios

8.5 Summary

The results of the three approaches to prediction in on demand and distance vector-based routing protocols has been studied. These are: Su's algorithms on prediction, the FNT approach described in this thesis, and a modified version of Su's algorithms to use the CM. A traditional approach to routing using a distance metric has also been used for comparison purposes.

The FNT + CM algorithms perform better than using no prediction for the purpose of increasing route stability. This has been observed in increased network connectivity as well as a reduction in routing time lost due to unexpected drop-offs. FNT + CM is able to predict on average 25% of handoffs accurately across all scenarios tested. FNT + CM also performs better than Su's traditional algorithm for route longevity, although drop-offs are higher due to the early prediction of Su's method. Su's algorithm by design will only predict RETs based on their task's current expiry time, reducing the extent of RETs. As predictions are recalculated as a new route is initiated, this has the effect of reducing the number of unexpected drop-offs, although at an increase of the number of routes established. The FNT + CM algorithms may increase the number of predicted handoffs if periodic updates of routes were to occur. The periodic updates of distance vector implementations, however, show that Su's methods are still able to predict more handoffs than the FNT design. In addition to this, the FNT is computationally more expensive to generate than Su's algorithms.

Adding a CM to Su's algorithm significantly improves its performance in all cases. With the aid of a CM and the short-duration nature of route predictions, Su + CM consistently predicts more handoffs and increases connectivity as a whole. In addition, Su's method + CM is better than FNT + CM in almost all experiments. In small scale networks, Su + CM consistently outperforms FNT + CM. Using FNT + CM selects longer routes, although the increased duration increases the chance of unexpected drop-offs. As RETs are not updated at periodic intervals in on demand-based routing protocols, and in distance vector-based protocols do not reconnected routes after termination, FNT + CM is at a disadvantage to the shorter RET approach of Su's algorithm. It is expected that with larger networks and more alternative routes, FNT + CM may select more stable routes because it can select longer routes, especially with FNT safety margins. This is suggested by FNT + CM producing the lowest number of routes established across

scenarios. Future work will be able to explore this further. Su + CM is recommended for small networks similar to those tested in this thesis.

It is not the purpose of this work to determine which of on demand and distance vector routing approaches is better. Distance vector-based protocols are shown to have reduced drop-offs, but at a cost of bandwidth and time lost while routes are unavailable until the next broadcast is received with a valid route. On demand-based protocols suffer more drop-offs, though the ability for the protocols to find new routes on demand increases connectivity. Ultimately, some form of hybrid approach would be beneficial, where route drop-offs can be re-established quickly, but where routes also receive regular updates on the quality of those routes. Such routing protocols have been proposed, but conclusive studies into their effectiveness has yet to be conducted, and is outside the scope of this research. The FNT and CM algorithms are designed to be easily used by any routing protocol, and as further advances are made in wireless ad hoc routing protocols, so too can FNT and CM results be improved. Overall the results of this chapter show that using FNT and CM algorithms as a form of prediction improves route stability and connectivity over traditional shortest-path approaches, though Su's approach combined with a CM is the recommended algorithm given the results found.

It is possible that routes could be selected on predicted highest signal strength, rather than longest expiry time, in both the FNT and Su's algorithms. This would select routes that are more stable. The current routing algorithms prefer longer duration routes over stronger signal routes, which is not ideal for connectivity. Introducing a greater safety margin this would have a similar effect, although any routes with borderline signal loss would then not be predicted as was evidenced with the safety margin experiments in this chapter. By using highest signal strength as a selection criteria, such borderline cases would still be predicted and have calculated RETs. Future research is left to explore this topic.

Chapter 9: Conclusions

In this thesis, a new approach to prediction in wireless ad hoc networks has been developed. This approach consists of two novel solutions, the Communication Map (CM) and the Future Neighbours Table (FNT). The CM is a real-time signal loss mapping solution. It is developed both without physical maps and without human intervention, using only mobile ad hoc nodes. To represent the CM, solutions with and without boundaries have been developed, though boundaries have been shown to have little effect on CM accuracy. The FNT provides detailed predictions of node connectivity to routing protocols. It analyses future connectivity between all pairs of nodes, using the CM to increase the accuracy of signal loss predictions. Two approaches, linear and polling, are used to develop the FNT based on the relationship between node tasks. The FNT is applied to existing metric-based routing protocols and results studied in the previous chapter.

The CM and FNT concepts are validated for use in improving connectivity in wireless ad hoc routing protocols over using no connectivity prediction. The CM itself has been shown to improve route stability in both the FNT and Su's algorithms. The FNT, however, does not outperform the contributions by Su [Su00]. In a number of studies, Su's algorithms presented results equal to or better than using the FNT. Su's algorithms are further improved when they are modified such that the CM can be applied to them. This may be largely due to the node population restraints of simulating such protocols. However, in the node populations studied, it is concluded that Su's algorithms with a CM produce better connectivity in small wireless ad hoc networks.

9.1 Contributions

This thesis has made a number of contributions to the field of using prediction in wireless ad hoc routing protocols. These are:

- A signal loss map solution has been developed (the CM). This signal loss map is the first to be developed in real time without human interaction. The CM is able

to automatically vary its level of detail. The CM is developed using a distributed approach, with each node in the network contributing signal loss information to the CMs of other nodes. The study of alternative CM models (i.e. cells and cells with boundaries) has also been studied, paving the way for further improvements.

- A novel approach to providing prediction information to routing algorithms has been developed with the concept of the FNT. This table provides a list of predicted connectivity states between a node and any other node in the network. Two approaches were studied in relation to developing a FNT: linear and polling. The linear approach was shown in Chapter 8 to produce faster results than polling, and with precise results based on the knowledge available. Significant limitations to the cases in which the linear approach may be applied were also identified for the first time. The polling approach, while not a significant contribution itself, was identified after the discovery of these limitations in the linear approach. These limitations exist but were not identified in the previously published work of [Su00]. The use of safety margins in FNT was also studied, though results for small-scale networks showed that they had a negative impact on prediction accuracy. This is due to not providing any prediction information on several routes where no alternate solutions were available.
- An extensive simulator was developed. This simulator was originally designed as a testbed for signal loss maps and prediction in wireless ad hoc networks, though the structure and design of it allows developments in a wide variety of directions via the component-based infrastructure of its architecture. The simulator provides native support for graphical design and modelling of wireless networks as well as signal propagation topology. Through the use of open standard XML input files, it is hoped that simulations can be interpreted and used in other wireless simulation tools. The collection of 12 statistical and 10 graphical views inbuilt into the simulator allows a quantitive analysis of various algorithms to be performed.
- The performance of several approaches to adding prediction to on demand and distance vector-based routing protocols was analysed. The results from Chapter 8 do not justify the use of FNT as an ideal prediction algorithm. However, the use of the CM in both FNT and Su's approaches to prediction was shown to improve performance of predictions.

9.2 Future Work

The CM has been designed to average signal loss propagation characteristics over a physical area. Two primary approaches were studied to achieve this goal; using cells to average signal loss, and using cells with outlining boundaries to average and identify signal loss. Various features, such as cell subdivisions and Time Block timings, have been studied in addition to the core CM design in an effort to produce a multi-purpose signal loss map. The open-ended nature of signal propagation environments leaves much room for improvement. Further study into signal loss maps is suggested:

- Another approach to averaging signal loss could be developed. Currently signal loss readings are averaged into Time Blocks, which are shared with other nodes. The Time Block approach supports sharing, but time-stamping signal loss readings into a larger single pool could be investigated. Alternatively, different windowing techniques could be investigated for use on Time Blocks.
- A different approach to averaging signal loss could be investigated, where signal loss readings are given a weight depending on their age, the number of nodes finding similar signal loss averages, or based on the likelihood of accuracy given the number of cells that the signal was estimated over.
- A more deterministic approach to signal loss mapping is another possibility. In such an approach, signal loss maps would be generated based on calculations of where signal loss may be. In this way signal loss is not averaged over areas, rather it is calculated based on remembering received signal loss readings and the estimated path they travelled. By recording signal loss readings and paths, it may be possible to calculate signal loss propagation areas and obstructions exactly. For example, nodes may determine a boundary of interference at an arbitrary location after learning that signals always travelling through a specific area have the same signal loss, regardless of distance travelled through that area. Cells in general would be calculated without any DCS, rather they would be discovered based on unusual changes to signal loss propagation when nodes travel around such areas. Such an approach is likely to be highly computationally intensive, but optimisations are also a topic of research.
- Signal loss map predictions in unmapped areas is a further area of study for future research. Unmapped cells may be able to predict initial signal loss estimates based on neighbouring cells. This may have a significant impact on

performance over the current model employed in the CM, where unmapped cells have an initial free-space signal loss modifier.

- Variations in node speed has not been considered. For a number of scenarios (for example, a mobile user on a train), the speed of the node will vary. The FNT algorithms assume a constant speed due to the complexities of any other model. It would be beneficial to study such complexities to determine their frequency in real life situations, and any adverse effects on performance they have.

During the testing of FNT in Chapter 8, a number of weaknesses were identified. In this thesis, neighbour connectivity times were found by identifying the times at which connectivity was predicted. To improve accuracy, safety margin values were introduced to lower the acceptable signal loss values that were used to establish connectivity. However, such an approach meant that even though some neighbours had connectivity, they no longer had predicted connection or disconnection events. An alternative approach may be to use signal strength as the link cost. Instead of merely predicting connectivity events, neighbours could have signal strength ratings over time, where periods of time could be classified such as "high signal strength", "low signal strength", "intermittent signal strength", etc. This way, if a route is available to routing algorithms but has an "intermittent signal strength" for the next minute, then routing algorithms may still select the route if it is the only one available, with the knowledge of how long signal strength may be sporadic. After the time period, signal strength may be predicted to improve, or may be predicted to drop off completely. In either case, providing this more detailed knowledge gives routing algorithms a greater chance of selecting the best routes for improving connectivity.

High node population testing was unable to be performed due to the computational and memory requirements of simulations. The development of improvements to simulator execution on multiple servers and in multi-processor systems is suggested to allow for a wider variety of experiments. Possible optimisations to the CM, FNT, and routing algorithms may also be possible to improve execution speed.

Previous to this research, only [Su00] had looked at predicting expiry times of connectivity for use in wireless ad hoc routing. Su modified existing routing algorithms such that the preferred routing metric was RETs instead of hop counts. Different

metrics were not studied in this research, but research could be continued into designing routing algorithms specifically to take advantage of predicted information. This would be even more beneficial where multiple information sources were available, such as the suggested FNT improvement of multiple time blocks of varying signal strength. Routing algorithms could be modified further so that as the CM changes, then updates of RETs could propagate the network. Because a route can constitute any number of links between two nodes, each pair of nodes would need to propagate updates to the potential expiry times of these links at regular intervals. This approach would create a hybrid routing algorithm based on both on demand and distance vector-based routing protocols. Also, more advanced routing protocols could be developed to fully use the FNT, including making use of future periods of connectivity, not simply the current connectivity's expiry time. RETs could also be used to dynamically alter the frequency of routing table broadcasts in distance-vector routing protocols.

In conclusion, this thesis presents methods for using prediction in wireless ad hoc routing protocols. The use of prediction improves connectivity and reduces unexpected drop-offs compared with traditional routing algorithms. Routing protocols can use prediction as a primary or secondary metric using the FNT, which bases predictions on future node mobility patterns combined with the CM, a signal loss mapping solution. These two solutions are designed to operate independently of user interaction to improve connectivity performance in automated wireless ad hoc networks.

Appendix A: Simulator Implementation

Design

The Wireless Ad hoc Network Simulator (WANS) is written in 20,095 lines of Java code across 114 classes. Java was selected because of its platform independence, where the possibility of moving algorithm code from a virtual simulator to real-world devices is facilitated by the fact that Java runs on almost any computing device. In principle, only small changes, replacing the simulated drivers with real ones, would need to be made in order to test the performance and viability of wireless communication protocols in real-world networks. The source code for the WANS project is available at: http://www.henrylarkin.com/phd/wans.zip.

The project is component-driven, divided into four major packages of code. These packages are Graphics, Protocol, Simulated and Statistics. This grouping separates each of the core abilities of the simulator. The Graphics package is responsible for designing and viewing simulations in progress. The Simulated package substitutes hardware devices with simulated software ones and is easily replaceable. The Protocol package contains all protocol implementations of the CM, FNT, and various routing protocols. Finally, the Statistics package provides algorithms and data structures for generating statistical information on simulated protocols.

Graphics

The Graphics package contains all code directly involved in creating and visually representing the network. These provide the interface with which the user interacts, and form the basis of the visualisation principle of the design goals of this project. This package is further segmented into four sub-packages: core graphics, components, frames, and map layers.

The Core Graphics sub-package consists of code for drawing individual objects. Many of these objects also allow for interaction by the user. As many of the objects revolve around shapes which are used in other algorithms for basic geometry computations, several classes include static code for this purpose. The classes overall include:

- GraphicsObject: An overall template from which all graphical objects are created. Examples of graphical objects include points, lines, rectangles, polygons, and maps of many objects.
- JMap: The foundation from which everything is drawn and interacted with. A Map consists of any number of Map Layers (a separate sub-package) onto which graphical objects are drawn.
- JScale: Code responsible for the scaling of real-world to visual coordinates, and vice-versa. All screen drawing methods require a JScale object as a method parameter.
- Paintable: A generic interface for any class which may be drawn on the screen.
- SLine: The basics of this class are for line drawing, though over 80% of the code is related to line algorithms such as intersections, finding points on line, and angles, which are used by both CM and FNT algorithms extensively.
- SMap: This class is used for objects which store many other objects, and is effectively a vector image, consisting of any number of other shapes.
- SPoint: This class represents a single point, and is used by all other shape-based classes for the points that are represented.
- SPolygon: This class represents a sequence of lines, which is used in drawing task lines based on each node's task list.
- SRectangle: The majority of CM objects and based around squares and rectangles, from which this class provides the foundation.
- STriangle: Represents triangle objects. Though no triangle drawing is used within the simulator, this class contains helper methods for calculating angles and line lengths, required by various FNT algorithms in determining whether two tasks may implement linear calculation or must use the polling method.

The Components sub-package contains two interaction panels used for the user interface:

- The Button panel: This panel provides a very basic interface for creating various buttons that can be related or stand-alone. It is used throughout the Monitor frame and the various map layers. The ButtonPanel class implements any grouping of buttons that may be required. The ButtonPanel is made up of 3 types of buttons, select buttons, toggle buttons and regular buttons. Select buttons are all grouped, in the sense that if one is depressed then the others are all raised. Toggle buttons can be up or down. Regular buttons act as normal single-click buttons.
- The Edit Object panel: This panel can be given any data object (for example, a single task, or the cell of a communication map), and display each of its data fields. It also has the ability to allow the user to enter changes to any of these fields, and to save them back to the original object. This panel provides the user with the ability to edit all details of simulation objects.

The Frames sub-package contains the designs of various windows used within the simulator. The three frames are:

- The Simulator frame: This frame provides control over the simulation, including speed and runtime. The scenario and settings files of a simulation can also be loaded and saved from this frame, as well as batch job execution.
- The Monitor frame: This window displays a simulation, and contains the various map layers and functionality for creating, simulating and viewing simulations.
- The Info frame: This is a basic template window for representing textual information. It includes functionality to automatically update itself with the latest textual information that it is displaying.

Map layers are a canvas onto which a certain type of information is displayed. They usually centre on a single task, for example one map layer might be responsible for designing the simulated environment. A map layer deals with actions performed on the layer (e.g. adding, moving, and deleting objects), as well as displaying the current representation of its design. Map layers include:

- The CM layer: This layer overlays the CM of a selected node over the network topology and allows information on any cell to be listed.
- The current clusters layer: Displays the logical grouping of nodes which are all able to transmit to each other.

- The tasking layer: Displays the future task path of a selected node. This layer also allows the task path to be modified or extended.
- The node layer: Displays all nodes in the network. This layer is used to add, remove, and position nodes in the network.
- The current neighbours layer: This layer displays lines of connectivity between all pairs of nodes which are in transmission range of each other.
- The simulated wireless map layer: This layer is used to design the simulated wireless environment through which nodes communicate.

Protocol

The Protocol package contains all classes related to protocol implementation and simulation. It is divided into five sub-packages: Core Protocol, Communication Map, Packets, Future Neighbours, and Routing.

The Core Protocol sub-package consists of classes which implement basic node functionality:

- Node: A class representing a single node through which all protocols execute. Under normal circumstances each node would have one hardware device to itself, but in the case of the simulator all nodes are simulated on a single hardware platform.
- Network Address: An encapsulation of the fields used to represent the network address of each node. In reality each node will have a unique identifier, but for the benefits of visualisation and interpretation in the simulated environment, the human tester can also give, as a network address, any textual name.
- Task: A class representing a future location that a node will travel to at a (continuously calculated) point in time.
- Task Line: A class representing a node's future travel between two Tasks.
- Tasking: The module responsible for maintaining and traversing through a list of future tasks for a single node.
- Current Neighbours: The module maintaining a list of nodes considered to currently be neighbours with another node.

- Future Neighbours: The module responsible for predicting which neighbours of a node will be active neighbours at future points in time.
- Generic Node: A wrapper class to represent either a node or a neighbour, providing abstraction to algorithms that do not differentiate between them.
- Neighbour: A class representing a neighbour of a node, along with current collected information about that neighbouring node.
- Neighbour Time Graph: The time graph representing the changes in status of a neighbour being neighbours with the current node over time. The time graph consists of Neighbour Time Graph Objects, which represent each individual status change.
- Signal Status Event: A data structure generated each time a signal is received by a node.
- Time: A class for representing and displaying the time in the simulation.

The Communication Map sub-package contains all classes implementing the CM algorithms described in Chapter 4:

- Averaging Time Window: A class used to regulate the average signal loss of a cell over time. The Averaging Time Window consists of Time Block objects.
- Boundary Coordinate: A data class representing the coordinate system of boundaries.
- Boundary Update Modifier: Contains the fields used to calculate and update the signal loss modifier of a single boundary based on the CM algorithms to map a received signal.
- Cell: Represents a single cell object as defined in the CM protocol.
- Cell Boundary: Represents a single boundary object as defined in the CM protocol.
- Cell Coordinate: A data class representing the coordinate system of cells.
- Cell Update Modifier: Contains the fields used to calculate and update the signal loss modifier of a single cell based on the CM algorithms to map a received signal.
- Communication Map: The module responsible for creating and maintaining a map of the perceived wireless environment.
- Communication Object: A wrapper class for cells and boundaries to be used by the modified CM algorithm which implements boundaries.

- Signal Line: Represents all information obtainable about a received signal, including the signal loss and assumed path of travel.
- Time Block: Implements the Time Block structure as defined in Chapter 4.
- Update Modifier: A wrapper class for cell and boundary update modifiers used when mapping received signals.

The Packets sub-package contains a collection of packet types which may be sent across the simulated network. These are all based on the Packet template, which defines the fields common to each packet:

- A unique packet id.
- The packet type.
- The source node address.
- The destination node address (or lack of one, indicating that the packet is a broadcast packet).
- The logical distance the packet travelled, calculated once the packet arrives at each destination. This information is used extensively with the next field to update the Communication Map.
- The line-of-sight signal line on which the packet was assumed to travel. Though in reality this is rarely the case, it is by far the easiest solution to implement, and in simulations all transmission calculations are done on this principle.
- The time that the packet was transmitted.

The various packet types that exist are:

- Ping Packet: A basic packet used to inform another node or nodes that a node exists.
- Map Update Packet: A packet used to encapsulate a node's CM updates.
- Current Neighbours Packet: A packet used to encapsulate the Current Neighbours table of a node.
- Global Broadcast Packet: A generic packet which will be propagated to all nodes in the network.
- Local Broadcast Packet: A generic packet which will be transmitted once and received by all neighbouring nodes within communication range of the transmitting node.

- Routable Packet: A generic packet which can be routed to any single node within the network.
- Routing Update Packet: A generic routing packet which details distance vector routing updates.
- Task Update Packet: A packet used to encapsulate the future task list of a node.

The Future Neighbours sub-package contains all algorithms and data structures to implement the FNT design presented in Chapter 5. These classes include:

- Future Neighbours: Contains all algorithms for generating a FNT.
- Future Neighbours Entry: Represents a single entry in the FNT.
- Future Neighbours Table: The FNT data structure and algorithms to find RETs.
- Neighbour Time Graph: A helper class for detailing the relationship between a pair of nodes over time.
- Real Task: A modified version of a Task specific for the FNT algorithms.
- Real Tasking: A modified version of the Tasking class specific for the FNT algorithms.
- SLOT Entry: Represents a single entry in a SLOT table.
- SLOT History: A data structure for storing previous SLOT tables for statistical purposes.
- Task Path: Represents an equalised list of tasks that a node will travel to at future points in time.

The Routing sub-package consists of all algorithms to execute the routing protocols presented in Chapter 6:

- DSDV: Implementation of the DSDV routing protocol.
- DSR: Implementation of the DSR routing protocol.
- DVMP: Implementation of the DV-MP routing protocol.
- FORP: Implementation of the FORP routing protocol.
- Route: A generic data structure representing a single route between a source and destination node.
- Route History: A data structure to collect previous routes for statistic purposes.
- Routing Algorithm: A generic template for routing protocols implementing common algorithms.

Simulated

The Simulated package contains classes which simulate devices and hardware interfaces that protocols rely on in order to function. It also includes code to execute simulations. The classes include:

- Simulator: The overall simulating class, which controls the speed of a simulation's time, as well as save and load algorithms.
- Node Simulator: Simulates the presence and existence of all nodes in a simulation.
- Node Simulator Settings: Provides a data store for the various settings required to simulate nodes.
- Network Simulator: Simulates the wireless network. This class represents a simulated wireless map of communication interferences which is used when determining which nodes will receive a transmitted packet from another node. A Simulated Network sub-package contains classes for simulated wireless propagation objects: Wireless Area, Wireless Boundary, Wireless Object, and Wireless Rectangle.
- Network Connection: A simulated network interface that allows each node to transmit packets to other nodes (controlled by the Network Simulator).
- Physical Control Simulator: Controls the movement of all nodes in the simulated network.
- Physical Control Module: Controls the speed and direction of travel of a node, which is controlled by a node's Task Path.
- Simulator Settings: This is the general configuration file for a simulation. All customisable properties of a simulation are contained in this data class (e.g. acceptable wireless signal (dBm), communication map update intervals).
- Batch Job: Contains algorithms for executing multiple scenarios over multiple settings.
- Random Node Settings: Contains algorithms for generating random node placement.
- Statistics: Provides code to extract data from all protocols in order to generate statistics from them.
- StatSet: Provides a data store of all the statistics being recorded for a simulation.

Statistics

The Statistics package contains a collection of all the statistic-generating algorithms for the various protocols implemented in the simulator. The purpose of the package is to provide data on overall algorithm performance, average over time graphs, and collations of all information over time. The package consists of seven classes: AverageStatisticSet, GraphDetails, GraphFactory, MultiTimeGraph, StatisticalTreeMap, StatisticSet, and TimeGraph.

File Formats

After several initial revisions of binary file formats for scenarios and settings, an XML-based file format was adopted. Each object which implements the Storable class interface has methods which allow the variables of an object to be saved and loaded with ease. Each variable is saved with a name, a type field, and the data. The name is usually the name of the variable, but can be named differently for easy grouping of many variables, such as in the example below (using "CM" to group Communication Map-related variables).

```
<CM.DEFAULT_CELL_SIZE type="java.lang.Integer">100</CM.DEFAULT_CELL_SIZE>
```

The type field is used to interpret the data, where primitive types are directly converted from text, and objects are recreated using Java's reflection abilities. Objects are represented in XML simply as nested XML tags, allowing for any depth of object to be stored. Originally there was the idea of representing each object once in separate tables, so that objects could be properly linked by multiple data entries without duplicate objects being generated after an object was reloaded from an XML file. However the data structures in the simulator were simple enough to not require any linking, and thus this feature of the file format was never finished. An example list of tasks for a node is shown in Figure 151. The average file size for XML-based scenarios and settings is between 10 and 30KB.

```
<tasks type="simulatethis.protocol.future_neighbours.RealTasking">
	<tasks type="libraries.structures.DataList">
		<element_0 type="simulatethis.protocol.future_neighbours.RealTask">
			<destinationLocation type="simulatethis.graphics.SPoint">
				<x type="java.lang.Double">30.0</x>
				<y type="java.lang.Double">10.0</y>
			</destinationLocation>
			<destinationTime type="java.lang.Long">44</destinationTime>
			<originLocation type="simulatethis.graphics.SPoint">
				<x type="java.lang.Double">20.0</x>
				<y type="java.lang.Double">30.0</y>
			</originLocation>
			<originTime type="java.lang.Long">0</originTime>
		</element_0>
		<element_1 type="simulatethis.protocol.future_neighbours.RealTask">
			<destinationLocation type="simulatethis.graphics.SPoint">
				<x type="java.lang.Double">150.0</x>
				<y type="java.lang.Double">100.0</y>
			</destinationLocation>
			<destinationTime type="java.lang.Long">344</destinationTime>
			<originLocation type="simulatethis.graphics.SPoint">
				<x type="java.lang.Double">30.0</x>
				<y type="java.lang.Double">10.0</y>
			</originLocation>
			<originTime type="java.lang.Long">44</originTime>
		</element_1>
		<element_2 type="simulatethis.protocol.future_neighbours.RealTask">
			<destinationLocation type="simulatethis.graphics.SPoint">
				<x type="java.lang.Double">30.0</x>
				<y type="java.lang.Double">70.0</y>
			</destinationLocation>
			<destinationTime type="java.lang.Long">591</destinationTime>
			<originLocation type="simulatethis.graphics.SPoint">
				<x type="java.lang.Double">150.0</x>
				<y type="java.lang.Double">100.0</y>
			</originLocation>
			<originTime type="java.lang.Long">344</originTime>
		</element_2>
		<listtype type="java.lang.String">java.util.LinkedList</listtype>
	</tasks>
</tasks>
```

Figure 151: Sample XML Task List

Using the Simulator

The simulator is controlled by two windows, the main simulator window (Figure 152, based on the Monitor Frame class) and the control window (Figure 153, based on the Simulator Frame class). In the centre of the main simulator window is the top-down view of the network and the nodes currently visible by it. The area that the map looks down on can be moved simply by dragging the background, and the mouse wheel can be used

to zoom in and out. The map interface is used primarily by the mouse, and depending on which layer is active, allows the user to view and modify that component of a simulation.

The map interface is controlled by the main view buttons. These are arranged vertically to the left of the map. These buttons allow the user to select which layer of the map is visible and active. Only one layer can be active at a time, though multiple layers may be toggled as display overlays. The active layer receives all mouse and keystroke actions. Some of these buttons do not select the active layer, but instead instantiate a window to provide textual information. These are the Info Frames that were discussed earlier. They provide a means of looking at raw data which is more useful when precise information is required.

Whenever a layer is selected, a panel of operations on this layer is provided in the lower-right of the main window. This panel usually consists of buttons which provide the actions that can be performed on the selected layer, such as adding a new node in the main node layer, or adding a wireless area or boundary in the simulated wireless environment map. This panel differs for each layer.

To the top-right of the main window is the selected object's panel. Whenever an object is selected (e.g. a node, a Communication Map's cell, a task point), it too may have actions or information with which the user may interact. Most objects have fields which the user may view and in some cases modify. These panels are based on the Edit Object Panel implementation. Some objects can also have actions associated with them, and these too are displayed as buttons below any object fields.

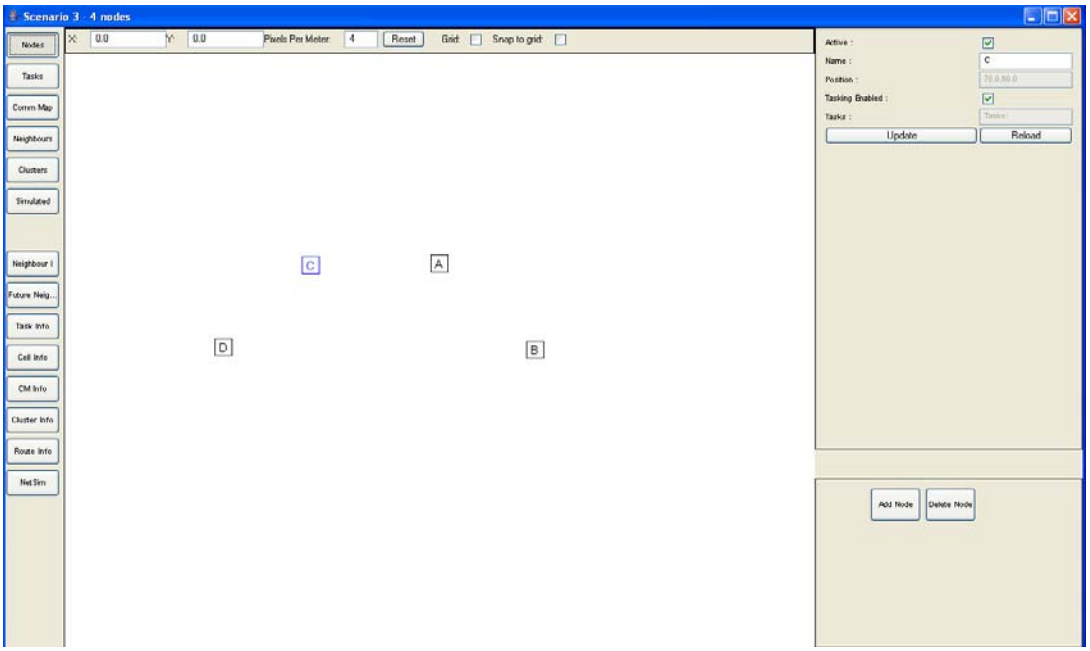

Figure 152: Main Window

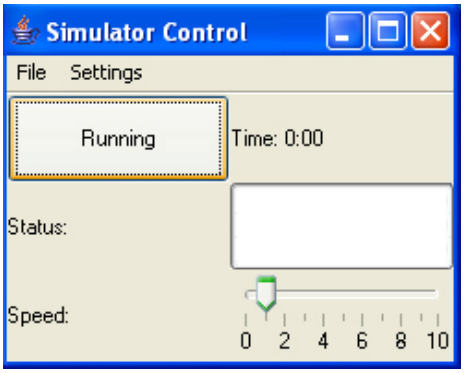

Figure 153: Control Window

The control window allows a simulation to be controlled. This window displays the current time in the simulated world, and has buttons to start and stop the simulator, and a slider to adjust the speed of time. Menus up the top of the main window provide the ability to create new simulations, as well as save and load simulations, and save all information to files. There are two options available for saving. One is to save a current snapshot at the current moment of simulation. The other is to auto-save snapshots at specific intervals. This latter option allows the user to get several snapshots of the simulation without having to wait for the required snapshot times and take each snapshot separately. Each snapshot consists of:

- The communication maps of all nodes (as images),

- The task paths of all nodes (as images)
- The simulated wireless map that is in use (as an image),
- A map of links between nodes showing which nodes are physically neighbours of other nodes (as an image),
- A diagram of the clusters that have been identified by a clustering algorithm, if one is being used (as an image),
- A document of statistics and settings used for the simulation (as a text file).

Simulator settings and statistic variables can also be edited through a pop up window. A screenshot of an example settings window is shown in Figure 154.

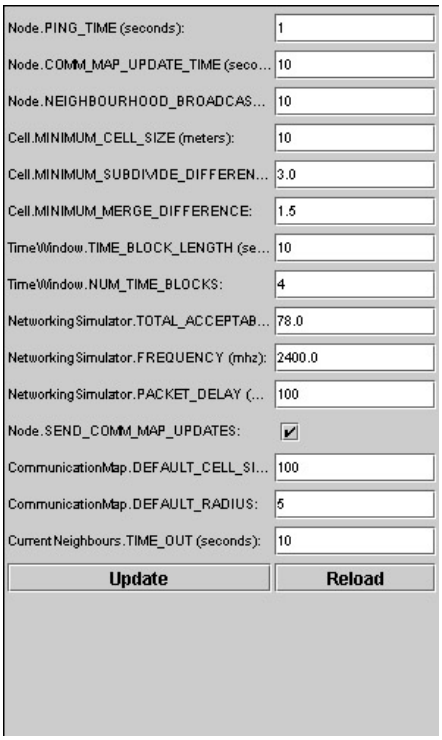

Figure 154: Settings Window

Tasking

The tasking layer allows the user to control the tasks that are issued to nodes. While a tasking model is not the focus of research (and thus does not require detailed simulation), it is necessary to represent and simulate some form of tasking to allow nodes to move with known destinations. This thesis focuses on automated wireless networks, where nodes have known future movements and destinations. It is necessary to simulate them

in order to test the FNT algorithms that rely on future node location information. Figure 155 shows a simple tasking interface, allowing node paths to be plotted and changed.

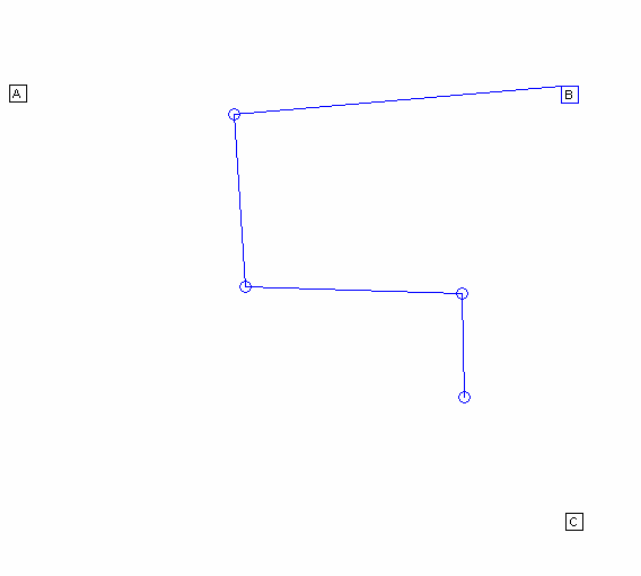

Figure 155: Plotting Tasks

At any stage during simulations, information on the task locations and expected arrival / departure times is also available. This allows the user to obtain specific details whenever any checking or specific documentation is required. Figure 156 shows an example of this.

Figure 156: Node Task Information

Simulated Wireless Map

One of the primary focuses of the simulator is on wireless signal strength mapping. The signals between nodes will vary depending on the simulated environment through which signals will pass. No complex signal modelling is done, only a direct line-of-sight algorithm is used. The simulated wireless environment can be made up of both *boundaries* and *areas*. *Boundaries* are barriers to communication transmission. They are once-off reductions of a given signal strength as a signal passes through them (e.g. a wall). *Areas* are an enclosed space where any signal travelling through that space has a higher signal

loss than in free space, depending on the modifier of that enclosed *area*. Both these structures allow a logical signal propagation environment to be created and used by the network simulating portion of the simulator, which deals with the transmission of packets via signals between nodes. An example simulated wireless map is shown in Figure 157. The map contains a structure with a signal loss modifier of 18.0 and a simulated boundary with signal loss of 150 logical metres (not shown).

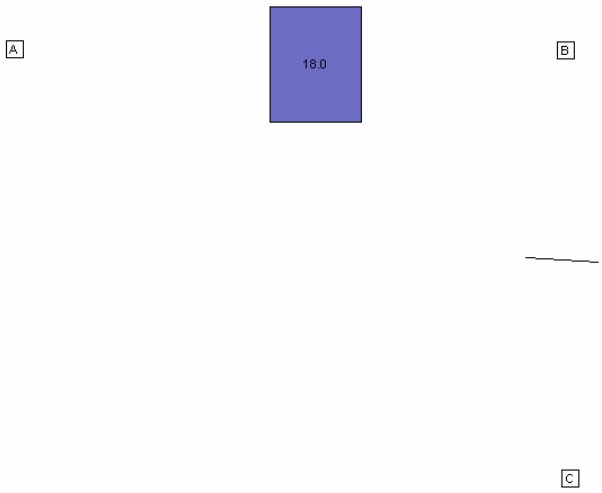

Figure 157: Sample Simulated Wireless Map

Communication Map

One of the most important views that the simulator provides is that of the CM of any particular node. This interface can be used to review the modifier values of each of the cells of a CM (and subdivided cells if a cell has subdivided). This map provides a quick method of reviewing the performance of the CM algorithms given any simulation (Figure 158). Combined with an overlay of the simulated communication map (Figure 159), the user can compare how a node views the signal propagation topology compared with the simulator's "reality". The CM layer of the simulator also allows the user to create a test signal between any two points. This can be used to find the CM's signal loss prediction as well as the simulated wireless map's actual signal loss value (Figure 160), both as a logical distance value and as a signal loss (dBm) value.

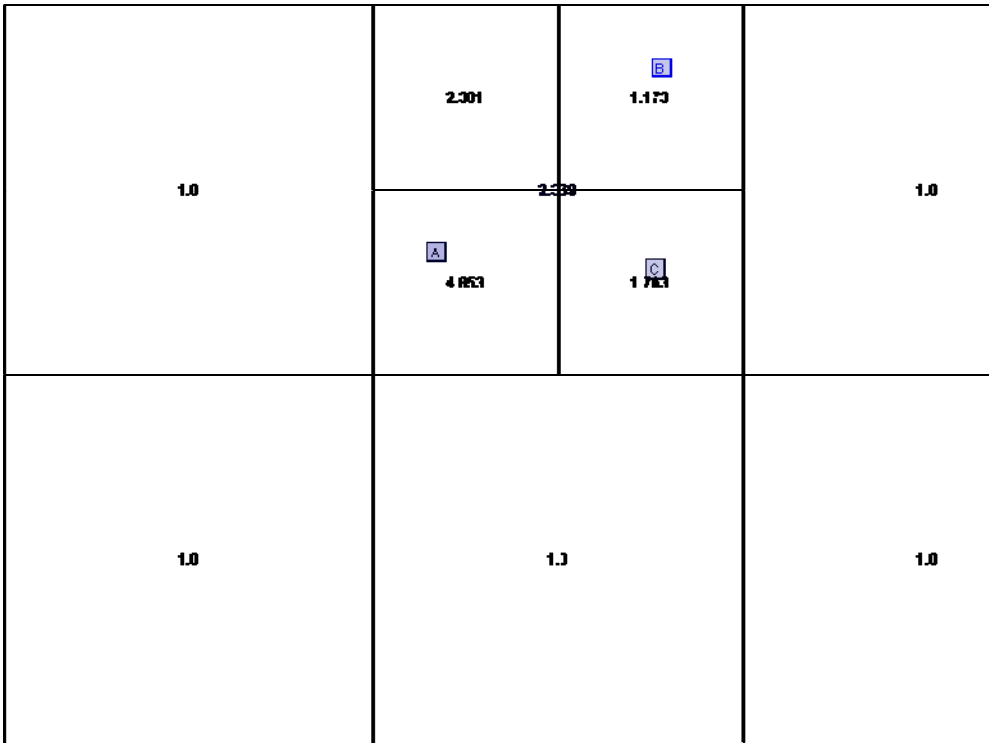

Figure 158: A sample Communication Map

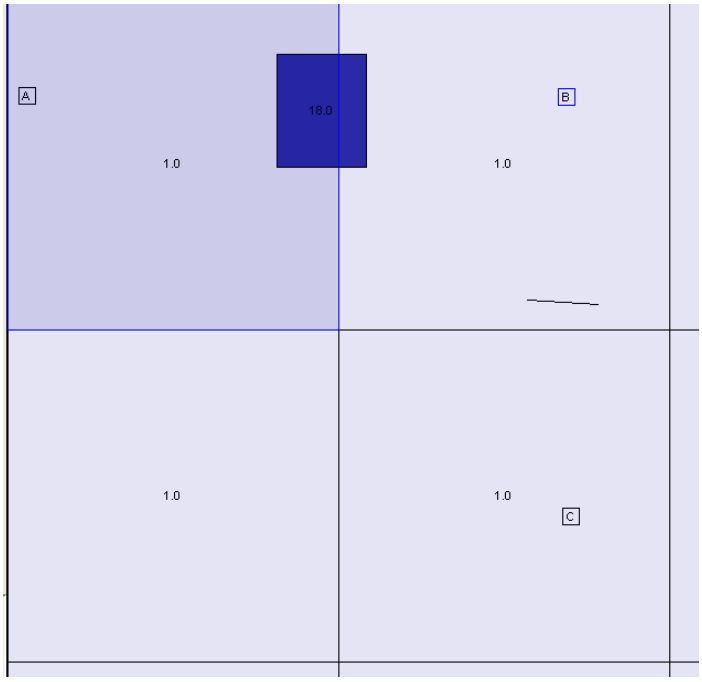

Figure 159: Overlay of the Simulated Wireless Map

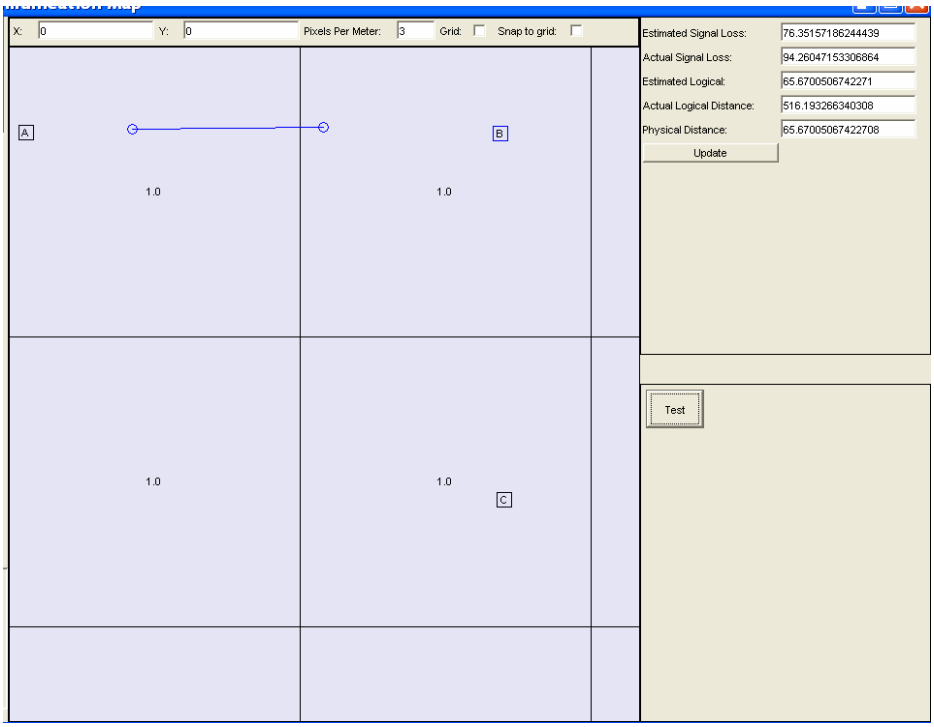

Figure 160: Creating a test signal

At any time, a cell may be analysed in detail by viewing its detailed statistics. These statistics show the signals received from each incoming packet, as well as any CM broadcasts from other nodes and how they have been calculated and contribute to the overall average for each time block since the node was first created. An example of this is shown in Figure 161.

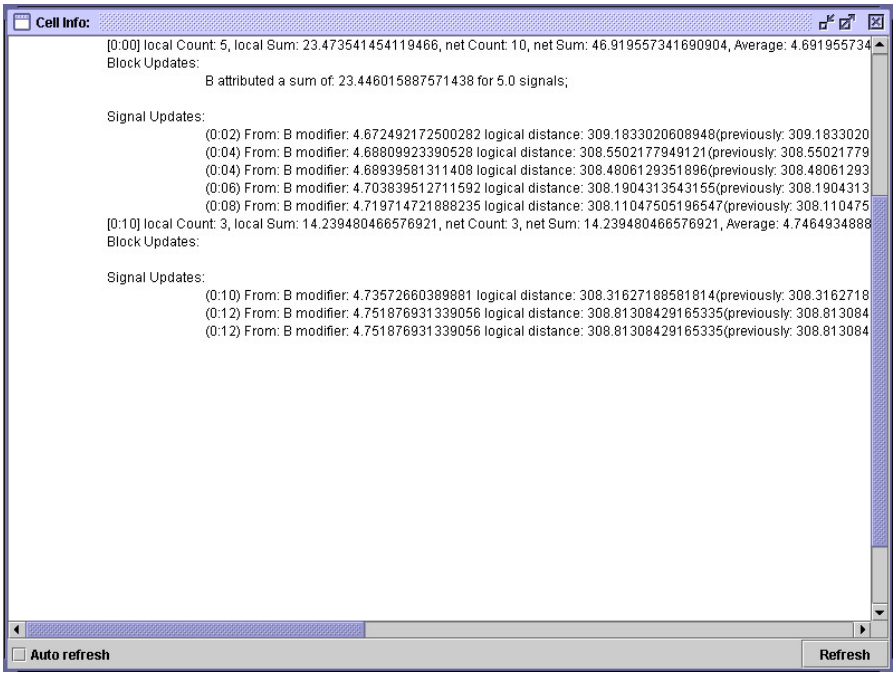

Figure 161: Sample Detailed Cell Information

Neighbours

The Current Neighbours Layer displays the connectivity between all nodes in the network. This is useful in understanding the effect of any simulated wireless map implemented. An example is shown in Figure 162.

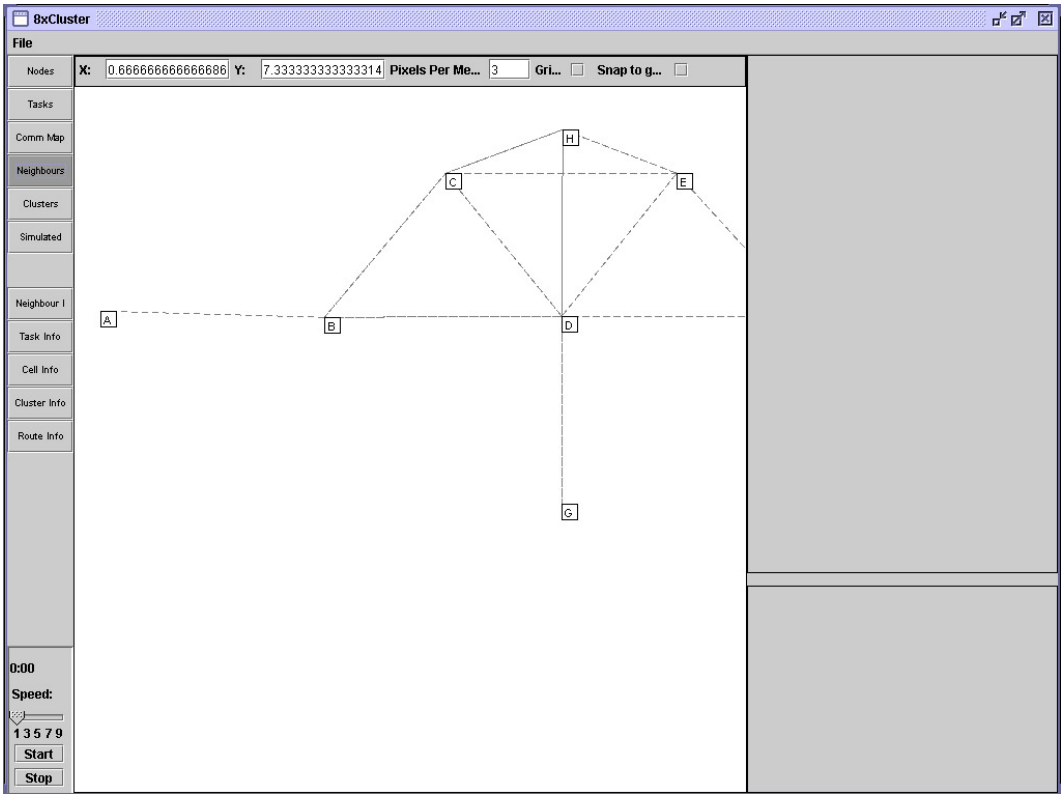

Figure 162: Sample Node Connectivity

Clusters

The Clusters layer provides a view to similar information to the Current Neighbours layer. The Clusters layer shows the logical groups of nodes all within transmission range of each other (Figure 163).

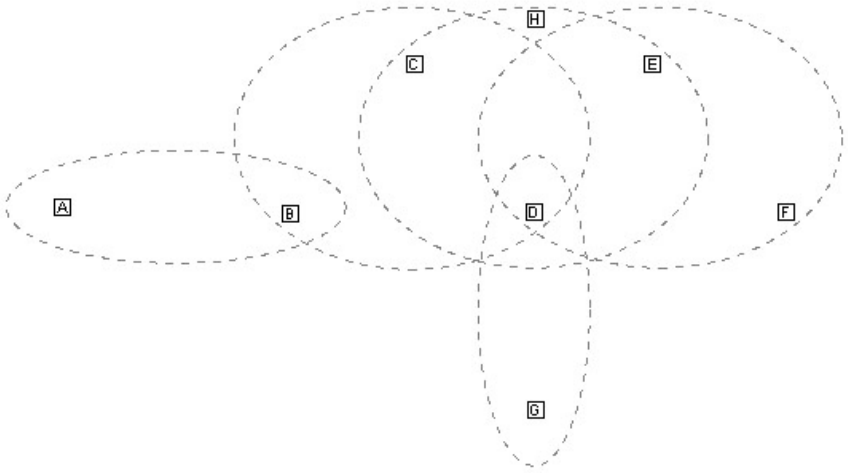

Figure 163: Sample Clustering

Summary

The purpose of the simulator is to aid testing and development of wireless ad hoc routing protocols based on a simulated wireless propagation map. This simulator provided the means to test the CM and FNT algorithms of this thesis, as well as compare the performance of a number of routing protocols implementing various aspects of the CM and FNT.

Appendix B: Default Settings used for Scenarios

Simulation Total Run Time	*1500 seconds*
Node.SEND_COMM_MAP_UPDATES	*true*
Node.IMPLEMENT_CLUSTERING	*false*
Node.PING_TIME	*1 seconds*
Node.COMM_MAP_UPDATE_TIME	*10 seconds*
Node.NEIGHBOURHOOD_BROADCAST_TIME	*10 seconds*
Node.FUTURE_NEIGHBOUR_RECALCULATE_TIME	*60 seconds*
Node.IMPLEMENT_FUTURE_NEIGHBOURS	*true*
Node.ROUTING_RECALCULATE_TIME	*30 seconds*
Node.IMPLEMENT_LINK_STATE	*true*
CommunicationMap.ALLOW_SUBDIVISIONS	*true*
CommunicationMap.IMPLEMENT_BOUNDARIES	*false*
CommunicationMap.MERGE_FOCUS	*false*
CommunicationMap.INACTIVE_SPREAD	*false*
CommunicationMap.DEFAULT_CELL_SIZE	*100 metres*
CommunicationMap.DEFAULT_RADIUS	*2 cells*
CommunicationMap.MINIMUM_CELL_SIZE	*10 metres*
CommunicationMap.MINIMUM_CELL_SUBDIVIDE_DIFFERENCE	*3.0*
CommunicationMap.MINIMUM_CELL_MERGE_DIFFERENCE	*1.5*
CommunicationMap.MINIMUM_BOUNDARY_SUBDIVIDE_DIFFERENCE	*20.0*
CommunicationMap.MINIMUM_BOUNDARY_MERGE_DIFFERENCE	*10.0*
CommunicationMap.HACK_GLOBAL_BROADCASTS	*true*
AveragingTimeWindow.TIME_BLOCK_LENGTH	*10 seconds*
AveragingTimeWindow.NUM_TIME_BLOCKS	*4*
NetworkingSimulator.TOTAL_ACCEPTABLE_LOSS	*85.0 dBm*
NetworkingSimulator.FREQUENCY	*2400.0 MHz*
NetworkingSimulator.MAX_BANDWIDTH	*2000000 bytes*
NetworkingSimulator.MAX_SIGNAL_VARIATION	*0.0 dBm*
NetworkingSimulator.SIGNAL_VARIATION_PROBABILITY	*0.0*
CurrentNeighbours.TIME_OUT	*10 seconds*
FutureNeighbours.SAFETY_MARGIN	*0.0 dBm*
FutureNeighbours.USE_COMMUNICATION_MAP	*true*
FutureNeighbours.POLL_INTERVAL	*1 second*
RoutingAlgorithm.USE_PREDICTION	*true*
RoutingAlgorithm.APPLY_FNT	*true*
RoutingAlgorithm.PREEMPTIVE_ROUTE_HANDOVER	*5 seconds*
RoutingAlgorithm.FORP.TIMEOUT	*2 seconds*

Appendix C: Glossary

ASL	Acceptable Signal Loss. The amount of loss (in dBm) required for a signal to be received by a neighbouring node.
CM	Communication Map. A logical map of signal loss over a physical space. It is developed and updated automatically by wireless ad hoc nodes.
DCS	Default Cell Size. The initial size of cells in the CM.
	Free-space loss. The propagation of a signal in an environment free of obstructions. The signal is neither amplified nor degraded due to the environment, but rather loses signal strength as it naturally propagates in all directions.
FNT	Future Neighbours Table. A table of future periods of connectivity between a pair of nodes.
LET	Link Expiry Time. The time at which a connection between two immediate nodes is predicted to terminate.
	Logical Distance. The distance that a signal would need to physically travel in free-space to produce an amount of loss.
	Handoff / handover. The process of changing the route used to send packets between a source and destination node from a less stable route to a more stable route.
MBMD	Minimum Boundary Merge Difference. The minimum difference between the highest modifier and lowest modifier of a boundary's subdivided boundaries needed for the two subdivided boundaries to merge back into a parent boundary.
MBSD	Minimum Boundary Subdivide Difference. The minimum difference between the highest modifier and lowest modifier of a boundary's subdivided boundaries needed for the boundary to subdivide.
MCMD	Minimum Cell Merge Difference. The minimum difference between the highest modifier and lowest modifier of a cell's subdivided cells needed for a group of subdivided cells to merge back into a parent cell.
MCS	Minimum Cell Size. The smallest size any cell in a CM can be subdivided to.
MCSD	Minimum Cell Subdivide Difference. The minimum difference between the highest modifier and lowest modifier of a cell's subdivided cells needed for a cell to subdivide.
MUP	Map Update Packet. A packet broadcast in the network to update the signal loss maps of all nodes.
RET	Route Expiry Time. The time at which a route between a source and destination node is predicted to terminate due to at least one link in the

	route disconnecting.
SLOT	Signal Loss Over Time table. A table used in the calculation of the FNT which details the times at which the rate of signal loss change between two nodes will change due to cells.
SWM	Simulated Wireless Map. A logical map delineating signal propagation characteristics over physical areas.

Bibliography

[AB25] Appleton, E., Barnett, M. (1925). *Signal Deviation in Transmission.* Proceedings Royal Society 109 A, pp. 621-624.

[ABO05] Argyros, A. A., Bekris, K. E., Orphanoudakis, S. C., Kavraki, L. E. (2005). *Robot Homing by Exploiting Panoramic Vision.* Autonomous Robots 19, Springer Science + Business Media, Inc., Netherlands, pp. 7-25.

[BGL00] Blazevic, Lj., Giordano, S. , Le Boudec, J. Y. (2000). *Self-Organizing Wide-Area Routing.* SCI 2000/ISAS 2000, Orlando, USA.

[BLS00] Bae, K. S., Lee, S., Su, W., Gerla, M. (2000). *The Design, Implementation, and Performance Evaluation of On-Demand Multicast Routing Protocol in Multihop Wireless Networks.* IEEE Network, Special issue on Multicasting.

[BPS06] Balakrishnan, H., Padmanabhan, V. N., Seshan, S., Katz, R. H. (1996). *A Comparision of Mechanisms for Improving TCP Performance over Wireless Links.* ACM Special Interest Group on Data Communications (SIGCOMM'96), California, USA, pp. 256-267.

[CDT02] Chatterjee, M., Das, S. K., Turgut, D. (2002). *WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks.* Cluster Computing 5, Kluwer Academic Publishers, pp. 193-204.

[CHH01] Capkun, S., Hamdi, M., Hubaux, J. (2001). *GPS-free positioning in mobile Ad-Hoc networks.* 34th Hawaii International Conference on System Sciences, Hawaii.

[CJBM01] Chen, B., Jamieson, K., Balakrishnan, H., Morris, R. (2001). *Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks.* ACM Mobile Computing and Networking, pp. 85-96.

[CJS03] Chellappa-Doss, R., Jennings, A. and Shenoy, N. (2003). *User Mobility Prediction in Hybrid and Ad Hoc Wireless Networks.* Australian Telecommunications Networks and Applications Conference (ATNAC'03), Australia.

[CWG97] Crow, B. P., Widjaja, I., Geun Kim, J, Sakai, P. T. (1997). *IEEE 802.11 Wireless Local Area Networks.* IEEE Communications Magazine, vol. 35, no. 9, pp. 116-126.

[DR92] Duchamp, D., Reynolds, N. F. (1992). *Measured Performance of a Wireless LAN.* 17th IEEE Conference on Local Computer Networks, pp. 494-499.

[EHH99] Estrin, D., Handley, M., Heidemann, J., McCanne, S., Xu, Y., Yu, H. (1999). *Network Visualization with the VINT Network Animator Nam.* tech. report 99-703, Computer Science Dept., University of Southern California, Los Angeles, USA.

[EM99] Enge, P., Misra, P. (1999). *Special issue on GPS: The Global Positioning System.* Proceedings of the IEEE, pp. 3–172.

[ES96] Eckhardt, D., Steenkiste, P. (1996). *Measurement and Analysis of the Error Characteristics of an In-Building Wireless Network.* ACM Special Interest Group on Data Communications (SIGCOMM'96), California, USA, pp. 243-254.

[Fal99] Fall, K. (1999). *Network Emulation in the Vint/NS Simulator.* IEEE International Symposium on Computers and Communications (ISCC'99), Egypt.

[FT02] Foka, A. F., Trahanias, P. E. (2002). *Predictive Autonomous Robot Navigation.* IEEE International Conference on Intelligent Robots and Systems (IROS'02), Lausanne, Switzerland.

[FTL95] Fritsch, T., Tutschku, K., Leibnitz, K. (1995). *Field Strength Prediction by Ray-Tracing for Adaptive Base Station Positioning in Mobile Communication Networks.* ITG Conference on Mobile Communication '95, Germany.

[GAP01] Goff, T., Abu-Ghazaleh, N. B., Phatak, D. S., Kahvecioglu, R. (2001). *Preemptive Routing in Ad Hoc Networks*. ACM Mobile Computing and Networking, pp. 43-52.

[HGP99] Hong, X., Gerla, M., Pei, G., Chiang, C.-C. (1999). *A Group Mobility Model for Ad Hoc Wireless Networks*. ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'99), Seattle, USA, pp.53-60.

[HJ01] Hu, Y., Johnson, D. B. (2001). *Implicit source routes for on-demand ad hoc network routing*. ACM International Symposium on Mobile Ad Hoc Networking and Computing.

[HKG01] Hong, X., Kwon, T., Gerla, M., Gu, D., Pei, G. (2001). *A mobility framework for ad hoc wireless networks*. ACM International Conference on Mobile Data Management (MDM'01), Hong Kong.

[HMS02] Howard, A., Mataric, M. J., Sukhatme, G. S. (2002). *An Incremental Self-Deployment Algorithm for Mobile Sensor Networks*. Autonomous Robots, Special Issue on Intelligent Embedded Systems. University of Southern California, USA, pp. 113-126.

[HP02] Hassan-Ali, M., Pahlavan, K. (2002). *A New Statistical Model for Site-Specific Indoor Radio Propagation Prediction Based on Geometric Optics and Geometric Probability*. IEEE Transactions on Wireless Communications, Vol. 1, No. 1, Jan 2002.

[HSS03] Howard, A., Siddiqi, S., Sukatme, G. S. (2003). *An Experimental Study of Localization Using Wireless Ethernet*. International Conference on Field and Service Robotics.

[JM96] Johnson, D. B., Maltz, D. A. (1996). *Dynamic Source Routing in Ad-Hoc Wireless Networks*. Mobile Computing, pp. 153-181.

[Kes88] Keshav, S. (1988). *REAL: A Network Simulator*. tech. report 88/472, Univ. California, Berkeley.

[KV98] Ko, Y., Vaidya, N. H. (1998). *Location-Aided Routing (LAR) in Mobile Ad Hoc Networks*. Texas A&M University. College Station, Texas, USA.

[KR01] Kubach, U., Rothermel, K. (2001). *Exploting Location Information for Infostation-Based Hoarding*. ACM Mobile Computing and Networking 2001 (July 16-21, 2001), University of Stuttgart. Stuttgart, Germany, pp. 15-27.

[LF98] Lynn, P. A., Fuerst, W. (1998). *Introductory Digital Signal Processing with Computer Applications*. Second Edition. John Wiley & Sons.

[LK00] Lee, S., Kim, C. (2000). *Neighbor supporting ad hoc multicast routing protocol.* ACM International Symposium on Mobile Ad Hoc Networking and Computing.

[LSG99] Lee, S., Su, W., Gerla, M. (1999). *Ad hoc Wireless Multicast with Mobility Prediction.* IEEE International Conference on Computer Communications and Networks (ICCCN'99), Boston, USA.

[LSH00] Lee, S., Su, W., Hsu, J., Gerla, M., Bagrodia, R. (2000). *A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols*. IEEE Conference on Computer Communications (Infocom'00), Tel Aviv, Israel.

[Luc] Lucent Technologies. http://www.lucent.com/.

[MG95] Murthy, S., Garcia-Luna-Aceves, J. J. *A Routing Protocol for Packet Radio Networks.* ACM International Conference on Mobile Computing and Networking, MOBICOM'95, pp. 86-95, 1995. California, USA.

[Mcc04] McClain, J. T. (2004). *A Behavior-Based Blackboard Architecture for Multi-Robot Control.* Masters Thesis. University of Georgia, Athens, Georgia.

[Mcd91] McDonald, C.S. (1991). *A Network Specification Language and Execution Environment for Undergraduate Teaching.* Proceedings of the ACM Computer Science Education Technical Symposium '91, San Antonio, Texas, pp. 25-34.

[Mcd93] McDonald, C.S. (1993). *Network Simulation Using User-level Context Switching*. Proceedings of the Australian UNIX Users' Group Conference '93, Sydney, pp. 1-10.

[MF53] Morse, P. M. and Feshbach, H. (1953). *Fourier Transforms*. Methods of Theoretical Physics, Part I, McGraw-Hill, New York, pp. 453-471.

[MFW04] Mauve, M., Fuler, H., Widmer, J., Lang, T. (2003). *Position-Based Multicast Routing for Mobile Ad-Hoc Networks*. Technical Report TR-03-004, Department of Computer Science, University of Mannheim, Germany.

[MG96] Murthy S., Garcia-Luna-Aceves J-J. (1996). *An efficient routing protocol for wireless networks*. ACM/Baltzer Mobile Networks and Applications 1996; 1: No. 2, pp. 183–197.

[MZ99] Mcdonald, A. B., Znati, T. (1999). *A Path Availability Model for Wireless Ad-Hoc Networks*. IEEE Wireless Communications and Networking Conference (WCNC'99), New Orleans, USA.

[MZ00] Mcdonald, A. B., Znati, T. (2000). *Predicting Node Proximity in Ad-Hoc Networks: A Least Overhead Adaptive Model for Selecting Stable Routes*. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'00), Boston, USA, pp.29-33.

[NKN96] Nguyen, G. T., Katz, R. H., Noble, B., Satyanarayanan, M. (1996). *A Trace-Based Approach for Modeling Wireless Channel Behavior*. Winter Simulation Conference (WSC'96), California, USA.

[OC96] Ortega, J. G., Camacho, E. F. (1996). *Mobile robot navigation in a partially structured static environment, using neural predictive control*. Control Engineering Practice, 4(12), 1996, pp. 1669-1679.

[Ono00] Ono, Y. (2000). *Corridor Navigation of a Mobile Robot Using A Camera and Sensors - Multi-Agent Approach*. Masters Thesis. University of California, Los Angeles, USA.

[OS89] Oppenheim, A.V., and R.W. Schafer (1989). *Discrete-Time Signal Processing*. Prentice-Hall.

[PB94] Perkins, C. E., Bhagwat, P. (1994). *Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers*. ACM Special Interest Group on Data Communications (SIGCOMM'94), London, UK, pp. 234-244.

[PC97] Park, V. D., Corson, M. S. (1997). *A highly adaptive distributed routing algorithm for mobile wireless networks*. IEEE Conference on Communications Society (INFOCOM'97), University of Maryland, USA, pp. 1405-1413.

[PGH00] Pei, G., Gerla, M., Hong, X. (2000). *LANMAR: Landmark Routing for Large Scale Wireless Ad Hoc Networks with Group Mobility*. Proceedings of IEEE/ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC'00), Boston, USA, pp. 11-18.

[PR99] Perkins, C., E., Royer, E., M. (1999). *Ad-Hoc On-Demand Distance Vector Routing*. 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), Louisiana, USA.

[RC98] Raleigh, G. G., Cioffi, J. M. (1998). *Spatio-Temporal Coding for Wireless Communication*. IEEE Transactions on Communications, vol. 46, no. 3, pp. 357-366.

[RT99] Royer, E., M., Toh, C-K. (1999). *A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks*. University of California / Georgia Institute of Technology. California / Atlanta, USA.

[Sha01] Shankar, P. M. (2001). *Introduction to Wireless Systems*. Wiley, USA.

[SHK04] Son, D., Helmy, A., Krishnamachari, B. (2004). *The Effect of Mobility-induced Location Errors on Geographic Routing in Ad Hoc Networks: Analysis and Improvement*

using Mobility Prediction. IEEE Wireless Communications and Networking Conference (WCNC'04), Atlanta, USA.

[SLG00] Su, W., Lee, S., Gerla, M. (2000). *Mobility Prediction and Routing in Ad Hoc Wireless Networks*. International Journal of Network Management, Wiley and Sons.

[SN02] Shah, S. H., Nahrstedt, K. (2002). *Predictive location-based QoS routing in mobile ad hoc networks*. IEEE International Conference on Communications (ICC'02), New York, USA.

[Su00] Su, W.W. (2000). *Motion Prediction in Mobile/Wireless Networks*. PhD dissertation. University of California, Los Angeles, USA.

[TDC01] Turgut, D., Das, S., Chatterjee, M. (2001). *Longevity of routes in mobile ad hoc networks*. 53rd IEEE Vehicular Technology Conference (VTS'01 Spring), Greece, pp. 2833-2837.

[THM95] Tadokoro, S., Hayashi, M., Manabe, Y. (1995). *On motion planning of mobile robots which coexist and cooperate with human*. IEEE International Conference on Intelligent Robots and Systems, pp. 518-523.

[Tuc93] Tuch, B. (1993). *Development of WaveLAN, an ISM Band Wireless LAN*. AT&T Technical Journal, vol. 72, no. 4, pp. 27-37.

[UAG94] Unger, B., Arlitt, M., Gburzynski, P., Gomes, F., Gurski, R., Ono-Tesfaye, T., Ramaswamy, S., Williamson, C. L., Zhong, X. (1994). *ATM-TN System Design*. WurcNet Inc. Technical Report.

[WL02] Wang, K., Li, B. (2002). *Group Mobility and Partition Prediction in Wireless Ad-hoc Networks*. IEEE International Conference on Communications (ICC'02), New York, USA, pp. 1017-1021.

[XB97] Xu, H., Brussel, H. V. (1997). *A behavior-based blackboard architecture for reactive and efficient task execution of an autonomous robot*. Robotics and Autonomous Systems 22, pp. 115-132.

[XHE01] Xu, Y., Heidemann, J., Estrin, D. (2001). *Geography-informed Energy Conservation for Ad Hoc Routing*. ACM Mobile Computing and Networking, pp. 70-84.

[YAS03] Youssef, M. A., Agrawala, A., Shankar, A. U. (2003). *WLAN Location Determination via Clustering and Probability Distributions*. IEEE International Conference on Pervasive Computing (PerCom'03), Texas, USA.

[YL01] Yu, F., Leung, V. C. M. (2001). *Mobility-based predictive call admission control and bandwidth reservation in wireless cellular networks*. IEEE International Conference on Computer Communications (INFOCOM'01), Anchorage, Alaska.

[YS01] Yousef, N. R., Sayed, A. H. (2001). *Detection of Fading Overlapping Multipath Components for Mobile-Positioning Systems*. IEEE International Conference on Communications (ICC'01), Helsinki, Finland, pp. 3102-3106.

[YS01b] Yousef, N. R., Sayed, A. H. (2001). *Adaptive Multipath Resolving for Wireless Location Systems*. Asilomar Conference on Signals, Systems, and Computers (ASILOMAR'01), California, USA.

[ZD97] Zonoozi, M. M., Dassanayake, P. (1997). *User Mobility Modeling and Characterization of Mobility Patterns*. IEEE Journal on Selected Areaas in Communications, vol. 15, no. 7.